Berufsakademie Sachsen
Staatliche Studienakademie Leipzig

Backup von MySql-Datenbanken

Praxisarbeit in der Fachrichtung Informatik
Im 2. Semester

Eingereicht von: Christian Gräfe
Erich-Köhn-Straße 53
04177 Leipzig
Seminargruppe: IT05
Matrikelnummer: 205830

Betreuer: Prof. Dr. habil. Harald Englisch
Beratung Medizin-Informatik
Burghausener Str. 18
04178 Leipzig, OT Böhlitz-Ehrenberg

Leipzig, 21.07.2006

Inhaltsverzeichnis

1. Einführung

Datenbanken spielen in der heutigen Zeit eine große Rolle in der Wirtschaft. Während sich früher hauptsächlich größere Unternehmen teure Datenbanksysteme leisten konnten, finden sich heute Datenbanken in irgendeiner Form in fast jedem Unternehmen. Sie werden zum Beispiel genutzt für die Finanzbuchhaltung, zur Dokumentenverwaltung, zur Pflege der Kundendaten oder für die Verwaltung von Lagerbeständen. Datenbanken stellen somit einen kritischen Teil jedes Unternehmens da. Fehler in der Datenbank oder gar Verlust der Daten können zu immensen wirtschaftlichen und finanziellen Schäden für das Unternehmen führen. Datensicherheit ist daher unerlässlich. Um auf Datenverluste schnell und verlässlich reagieren zu können, ist das Erstellen von Backups unumgänglich. Die richtige Backupstrategie sorgt nicht nur bei normalen Daten für hohe Sicherheit gegenüber Datenverlusten, sondern auch bei Datenbanken.

Aus persönlicher Erfahrung kann ich sagen, dass das Vernachlässigen von Backups schnell zu schlaflosen Nächten führen kann. Die Daten in einer Datenbank lassen sich nicht so ohne weiteres ohne ein aktuelles Backup wiederherstellen. Gelöschte Dateien auf einer Festplatte oder einem ähnlichem Datenträger lassen sich meist mit geringem Aufwand rekonstruieren. Gelöschte Datensätze in einer Datenbank sind für immer verloren, wenn es keine Sicherheitskopie gibt. Wichtig ist vor allem aber auch das Alter des Backups. Ist die einzige Sicherung einer Datenbank, an der jeden Tag gearbeitet wird, ein drei Monate altes Backup, weist das auf eine unzureichende Backupstrategie hin.

Diese Praxisarbeit befasst sich im ersten Teil mit der Notwendigkeit von Backups und den unterschiedlichen Backupstrategien und -formen für Datenbanken im allgemeinem. Im zweiten Teil der Praxisarbeit wird besonders auf das freie Datenbanksystem MySql eingegangen. Da MySql heutzutage große Verbreitung auf Webservern findet, wird auf die verschiedenen Methoden für das Erstellen eines Backups einer Online-Datenbank eingegangen.

2. Notwendigkeit von Backups

Generell kann man zwei Arten unterscheiden wie es zu Datenverlusten kommen kann. Durch interne Einflüsse und externe Einflüsse. Bei internen Einflüssen gibt es 3 Arten von Fehlersituationen bei denen es zu Datenverlust kommen kann.

- Transaktionsfehler
- Systemfehler
- Mediafehler

Transaktionsfehler (engl. transaction failure) treten bei Transaktionsabbruch durch das System bzw. durch den Benutzer auf. Diese Fehlerart wird auch als lokaler Fehler einer Transaktion bezeichnet, da diese Fehler keinen Einfluss auf das restliche System haben.
Systemfehler (engl. system failure) sind alle Fehler im Zusammenhang mit dem Datenbank-Management-System (DBMS), dem Betriebsystem oder der Hardware. Bei ihnen kommt es zum Verlust der Daten im flüchtigen Speicher.
Mediafehler (engl. media failure) bezeichnet alle Fehler, die den Verlust von Datenbankdaten im stabilen Speicher nach sich ziehen. Häufige Ursachen sind:

- „Head-Crashes“, die Teile des stabilen Speichers, z.B. einigen Plattensektoren, beschädigen
- Controller-Fehler, die zu Datenverlusten führen
- Naturgewalten wie Feuer, Erdbeben oder Hochwasser, wodurch der stabile Speicher zerstört werden kann

Der Datenverlust bei Transaktions- und Systemfehlern wird durch das DBMS minimiert, wenn nicht sogar ganz vermieden (für nähere Informationen siehe [1]). Gegen Datenverlust bei Mediafehlern kann nur durch das regelmäßige Archivieren der Daten auf einem vom System unabhängigen Speichermedium, zum Beispiel CD oder DVD, vorgebeugt werden.

Als externe Einfluss ist der Faktor Mensch anzusehen. Durch den Benutzer des DBMS kann es zu gewollten oder ungewollten Datenverlusten kommen. Fehlerhafte bzw. unüberlegte Lösch- oder Updateanfragen an das System lassen sich bei den meisten DBMS nicht rückgängig machen. In so einem Fall hilft nur eine vorangegangene Datensicherung weiter, um die ursprünglichen Daten wiederherzustellen.

3. Backupstrategien

Ein Backup ist eine Kopie eines konsistenten Datenbankzustandes. Üblicherweise werden Backups nach einem bestimmten Zeitplan erstellt, etwa am Ende von Arbeitstagen oder an Wochenenden.

Grundsätzlich unterscheidet man zwei Zustandsarten der Datenbank während des Erstellen des Backups: zum einen „HotCopy“ und zum anderen „ColdCopy“.
Als *HotCopy* bezeichnet man den Backupvorgang, während weiter mit der Datenbank gearbeitet wird. Die Datenbank steht somit für Anfragen zur Verfügung. HotCopy findet vor allem in Bereichen Anwendung, wo das Herunterfahren des DBMS aus technischen oder zeitlichen Gründen nicht möglich ist bzw. das Sperren der Datenbank laufende Prozesse gefährdet.
ColdCopy ist nun das genaue Gegenteil zum HotCopy. Bei der ColdCopy ist kein Arbeiten mit der Datenbank während der Zeit der Backuperstellung möglich. Bei der ColdCopy läuft das DBMS nicht bzw. die Datenbank, oder Teile von ihr, sind durch das DBMS gegen Zugriffe gesperrt. (vgl. MySql „LOCK TABLES“ „UNLOCK TABLES“)

Bei der Lagerung des Backups ist zu beachten, ob es sich um eine Datensicherung oder eine Datenarchivierung handelt. Auch wenn die zwei Begriffe häufig synonym verwendet werden, sind es doch zwei unterschiedliche Dinge.
Bei *Datensicherung* ist ein kurz- bis mittelfristiger Erhalt der Daten gemeint. Die Datensicherung ist für die Wiederherstellung der Daten im laufenden Prozess gedacht, also um verlorene Daten so qualitativ wie möglich zu rekonstruieren und das System am laufen zu halten.
Unter *Datenarchivierung* versteht man das langfristige Erhalten von Datenbankzuständen eines bestimmten Zeitpunktes. Die Datenarchivierung erfolgt mit dem Zweck, den Prozess der Datenänderung über einen längeren Zeitraum nachzuvollziehen.

Für die Datenarchivierung wird immer eine vollständige Kopie der Datenbankdaten benötigt, bei der Datensicherung ist dies nicht der Fall. Hier ergeben sich durch Häufigkeit des Erstellens und die Anzahl der zu speichernden Daten folgende Backuparten:

- Vollständige Datensicherung
- Differenzielle Datensicherung
- Inkrementelle Datensicherung

Eine v*ollständige Datensicherung* oder auch Komplettbackup ist das vollständige Abbild der Datenbank zu einem bestimmten Zeitpunkt. Dies ist in der Regel sehr einfach zu bewerkstelligen und wird im Abschnitt 4.2 näher erläutert, das Zurückspielen der Daten gestaltet sich ebenfalls recht einfach. Nachteil der Komplettsicherung ist die Dauer des Erstellens. Gerade bei großen Datenbanken kann dies mitunter sehr lange dauern. Erfolgt das Backup als „Cold Copy“ kann in dieser Zeit nicht mit der Datenbank gearbeitet werden.

Die *differenzielle Datensicherung* findet sich auch unter dem Namen Delta-Methode. Um Platz und Zeit zu sparen, werden hierbei nur Daten gesichert, die seit dem letzten Komplettbackup geändert wurden. Bei der Rücksicherung der Daten wird dann zusätzlich noch dieses Komplettbackup benötigt.

Die i*nkrementelle Datensicherung* ist der differenziellen sehr ähnlich. Bei der inkrementellen Datensicherung werden ebenfalls nur die geänderten Daten gesichert, allerdings bezogen auf das letzte inkrementelle Backup. Hierbei kann allerdings das letzte inkrementelle Backup auch ein vollständiges Backup sein.

Die inkrementelle Datensicherung erfolg meist in mehreren Ebenen. Unter Ebene 0 versteht sich das vollständige Backup der Datenbank. Ebene 1 bis N sind inkrementelle Datensicherungen der Datenbank, wobei Ebene N die Änderung seit Ebene N-1 beinhaltet. Daraus ergibt sich, dass für das Wiederherstellen der Datenbank N+1 Backups eingespielt werden müssen. Diese Variation der Datensicherung findet in der Form Anwendung, dass zum Beispiel am Wochenende ein vollständiges Backup erstellt wird und an den Wochentagen ein inkrementelles Backup. Bei 5 Arbeitstagen hätte man somit 6 Ebenen der Datensicherung. [2,3]

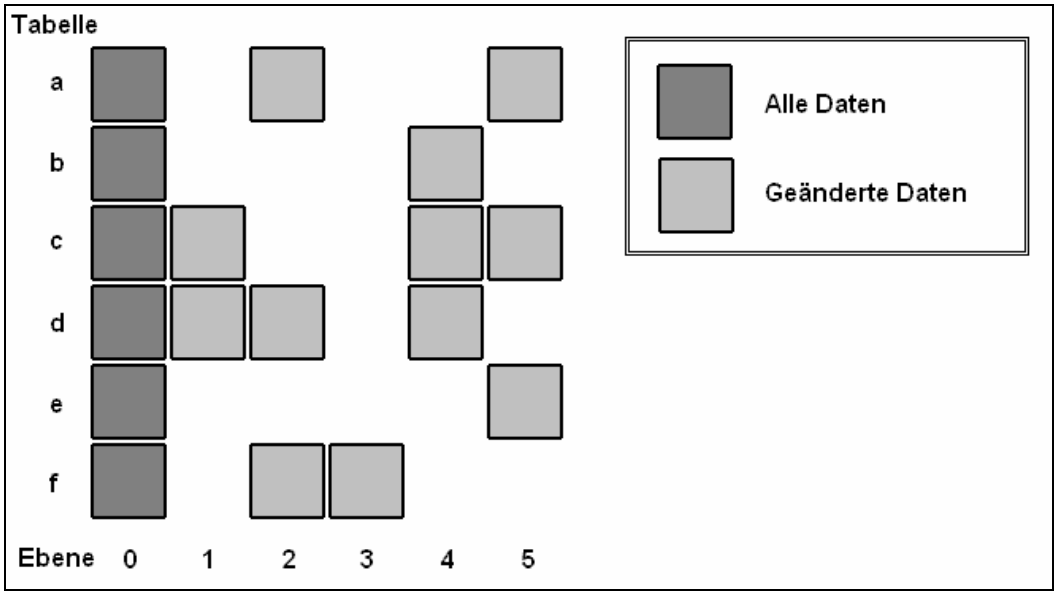


Abbildung 1 - Inkrementelles Backup mit mehreren Ebenen

4. MySql – Was ist das?

4.1 Aufbau und Einsatz des MySql DBMS

MySql ist ein von der Firma MySql AB („AB“ schwedisch für Aktiengesellschaft) entwickeltes Datenbankmanagementsystem. MySql gehört zu den am weitesten verbreiteten Open-Source-Programmen. Mit mehr als 10 Millionen aktiven Installationen wird es sowohl für große Unternehmenssysteme als auch für ganz spezielle integrierte Anwendungen eingesetzt. MySql findet vor allem in Zusammenhang mit dem Webserver Apache und PHP Verwendung. Je nach Hostbetriebssystem wird diese Kombination LAMP (Linux Apache MySql PHP) oder WAMP (Windows Apache MySql PHP) genant. Durch diese Verbindung finden MySql-Datenbanken häufig Verwendung bei Internetseiten und Webanwendungen. Ein großer Vorteil des MySql DBMS ist die Verwendung von verschiedenen Tabellentypen wie zum Beispiel MyISAM, InnoDB oder Berkley DB. Es können auch eigene Engines eingebunden werden. Für Webseiten wird überwiegend die MyISAM-Engine genutzt. [4]

MySql unterstützt theoretisch eine unbegrenzte Anzahl an Datenbanken mit jeweils beliebig vielen Tabellen, welche unter Verwendung der MyISAM-Engine jeweils eine maximale Größe von 64PB (ca. $64*10^{15}$ Byte) haben dürfen. Nur Theoretisch allerdings, da die Größe durch das Betriebssystem beschränkt wird. MySql speichert für jede Tabelle drei Dateien, eine Definitionsdatei (.frm), eine Indexdatei (.MYI) und eine Datei mit den Tabellendaten (.MYD). Liegt die Datei mit den Tabellendaten auf einem FAT-Dateisystem, kann die Tabelle aufgrund der Beschränkung der Dateigröße durch das System eine Größe von 2 GB nicht überschreiten. Auf FAT32 Systemen sind 4 GB möglich, die MySql AB rät allerdings beim Produktionseinsatz des DBMS von der Verwendung des FAT-Dateisystems ab. Laut Angaben der Mysql AB[1] sind mit NTFS Tabellengrößen von 2 TB möglich, mit Solaries 9/10 sogar bis zu 16 TB. Unter Verwendung des ext3 Dateisystems unter Linux (2.4+) sind 4 TB möglich.

[1] http://dev.mysql.com/doc/refman/5.1/de/table-size.html

Die Arbeit mit der Datenbank erfolgt über das Client/Server- Model, wobei Client und Server räumlich getrennt sein können. Die MySql-Anbindung eines Webservers zum Beispiel kann über das Internet von einem Arbeitsplatzrechner aus gesteuert werden. Der Einsatz als Embedded-Server in einem System ist allerdings auch möglich.

Der Clientzugriff auf eine MySql Datenbank gestaltet sich recht einfach, da MySql die gängigen Verbindungstypen ODBC und JDBC unterstützt. Die Nutzung der verschiedenen Anbindungen an Programmiersprachen, wie zum Beispiel der C-API, der PHP-API oder der PERL-API, gestaltet sich durch die gute Dokumentation der Schnittstellen als recht einfach.

4.2 Verschiedene Wege zum Backup einer MySql-Datenbank

Wenn viele Wege nach Rom führen, führen auch viele Wege zum Backup einer MySql-Datenbank. Generell kann man zwei Herangehensweißen bei der Erzeugung des Backups unterscheiden. Zum einen natürlich das Kopieren der Tabellendaten direkt aus dem Datenbankverzeichnis heraus, welches per direkten Zugriff oder mit Hilfe von speziellen Programmen erfolgen kann. Und zum anderen das Abfragen der zu speichernden Daten über SQL-Abfragen und anschließendes Speichern der Abfrageergebnisse in einer neuen Datei.

Die einfachste Möglichkeit ist das direkte Kopieren der Tabellendaten aus dem Datenbankverzeichnis heraus. Damit es bei diesem Vorgang nicht zu Komplikationen kommt, muss der Zugriff auf die zu kopierende Tabelle gesperrt werden und nach dem Kopieren wieder freigegeben werden. Dies geschieht mit den folgenden MySql-Abfragen:

- „LOCK TABLES tablename“ zum Sperren der Tabelle „tablename“
- „UNLOCK TABLES“ zum Freigeben der Tabelle

Dabei kann „tablename“ auch eine Liste durch Komma getrennter Tabellennamen sein, um mehre Tabellen gleichzeitig zu sperren. Nach dem Sperren der Tabelle sollte noch ein „FLUSH TABLES“ ausgeführt werden, damit noch im Cache befindliche Daten in die Datenbank geschrieben werden. Ist die Tabelle einmal gesperrt, können die Daten problemlos kopiert werden. Um diese ganzen Schritte nicht jedes Mal selber per Hand vornehmen zu müssen, gibt es bei MySql ein vorgefertigtes PERL-Skript namens „mysqlhotcopy“[2]. Nachteile sind allerdings, dass es nur auf dem Computer ausgeführt werden kann, auf dem auch das Datenbankverzeichnis gespeichert ist und dass es nur unter Unix und NetWare läuft.

Wenn der direkte Zugang zum Datenbankverzeichnis nicht gegeben ist, bietet sich die Möglichkeit, die Daten mittels SQL-Abfragen auszulesen. Die Ausgabe der folgenden zwei Abfragen enthält alle nötigen Informationen, um die Tabelle zu einem späteren Zeitpunkt wiederherzustellen.

- „SHOW CREATE TABLE tablename“ gibt die Struktur der Tabelle aus
- „SELECT * FROM tablename“ liefert den gesamten Inhalt der Tabelle

Eine ähnliche Ausgabe erzeugt auch das Programm „mysqldump“[3].

---

[2] http://dev.mysql.com/doc/refman/5.1/de/mysqlhotcopy.html

[3] http://dev.mysql.com/doc/refman/5.1/de/mysqldump.html

5. Praxisbeispiel: Backup einer Online-Datenbank

5.1 Vorbetrachtungen

Bei diesem Beispiel ist das Backup einer etwa 10 MB großen MySql-Datenbank mit ca. 50 Tabellen beabsichtigt. Es gibt Tabellen, die sich täglich ändern, und Tabellen, die sich selten, also weniger als einmal die Woche, ändern. Die Datenbank wächst etwa um 100KB pro Woche.

Die Datenbank liegt auf einem Webserver, auf dem LAMP zum Einsatz kommt. Ein direkter Zugriff auf das Datenbankverzeichnis (das Verzeichnis, in dem MySql die Datenbanken/Tabellendaten speichert) steht nicht zur Verfügung. Des Weiteren existiert ein FTP-Zugang zum Home-Verzeichnis auf dem Webserver.

Der Traffic ist auf 6GB pro Monat beschränkt. Die Datenbank wird für das dynamische Erstellen von Webseiten über PHP-Skripte genutzt. Die Seitenaufrufe belaufen sich durchschnittlich auf etwa 300 bis 400 pro Tag. Da die Seite sehr schlicht gehalten ist, kann mit einem Traffic pro Aufruf von 500 Kilobyte gerechnet werden. Somit ergibt sich bei 30 Tagen und 350 Aufrufen ein Traffic von 5250 MB. Für restliche Arbeiten an der Seite und für das Backup der Datenbank ergibt sich somit ein Resttraffic von 750MB. Somit stehen pro Tag etwa 25MB Traffic zur Verfügung. Geht man nun davon aus, dass für die tägliche Pflege der Seite 5MB Traffic, durch das Einstellen von neuen Nachrichten oder Bilderupload, verbraucht werden, ergibt sich für das Backup ein maximal möglicher Traffic von 20MB.

Bei dieser Größe ist ein tägliches vollständiges Backup zwar möglich, aber nicht empfehlenswert und nicht zukunftsträchtig. Durch das ständige Anwachsen der Datenbank und der Seite an sich, werden diese 20MB Traffic immer weiter eingeschränkt, somit wird es nach etwa einem Jahr nicht mehr möglich sein ein tägliches Komplettbackup der Datenbank zu erstellen. Aus diesem Grund muss entweder der Monatstraffic erhöht werden, was allerdings Kosten nach sich zieht, oder von täglichen Komplettbackups auf tägliche inkrementelle Backups umgestellt werden.

5.2 Umsetzung in einem Programm

Zur Umsetzung des im vorangegangenen Kapitel geschilderten Problems bot sich die Umsetzung in einem PHP-Skript an, da die Verbindung zwischen MySql und PHP sehr gut ausgebaut ist. Es trat allerdings schon sehr bald ein sehr gravierender Nachteil dieser Lösung in Erscheinung: Da ein PHP-Skript nur eine bestimmte Ausführungszeit haben darf, meistens 30 Sekunden, der Download der Tabellendaten allerdings viel länger dauert, wurde diese Lösung schnell verworfen. Es wäre zwar auch möglich, die Daten vorher auf dem Server komplett in einer Datei zu sichern und diese dann zum Download anzubieten, da die Lösung aber so einfach und effektiv wie möglich sein soll und das Downloaden dieser Datei unnötigen Traffic verursacht, wurde diese Variante nicht gewählt.

Da MySql über eine sehr gut dokumentierte[4], leicht verständliche und sehr effektive C-API verfügt, kommt nun als Backuplösung ein eigenes C++ Programm zum Einsatz.
Forderungen an das Programm waren:

- Vollständiges Backup möglich
- Differenzielles Backup möglich
- „Ein-Klick-Start“, d.h. das Programm kümmert sich selber um alles (Verbindungsdaten über Ini-Datei einlesen)
- Geringer Traffic

Um das Zurückspielen der Daten so einfach wie möglich zu halten, wird bei jedem Durchlauf des Programms ein vollständiges Backup erzeugt. Da der Traffic aber so gering wie möglich gehalten werden soll, erfolgt das Erzeugen des vollständigen Backups nicht nur mit Daten aus der Datenbank, sonder auch mit den Daten aus dem vorangegangenen Backup. Ist kein solches Backup verfügbar, zum Beispiel beim ersten Start des Programms, wird ein vollständiges Backup aus der Datenbank heraus erzeugt.
Der einzige Nachteil des Ganzen ist die Größe der Backupdatei, da sie immer die Größe eines vollständigen Backups hat.

[4] http://dev.mysql.com/doc/refman/5.1/de/c.html

Im Anhang findet sich ein grober Programmablaufplan (PAP) des Programms und die Internetadresse, unter dem der Quellcode und das Programm (für Windows) zu finden ist. Aufgrund der Einfachheit und relativ schnellen Ausführung trägt das Programm den Namen „FastMySqlBackup“. Das Programm ist voll funktionsfähig und läuft zurzeit unter Windows. Da keine betriebssystembedingten Funktionen verwendet werden, sollte es auch unter Linux und anderen Betriebssystemen laufen, auf denen die MySql Laufzeitbibliotheken zur Verfügung stehen.

Für zukünftige Versionen des Programms ist eine Aufteilung in mehrere Threads denkbar. Das Abfragen der Daten aus der Datenbank und das Kopieren aus dem vorangegangenen Backup können zeitgleich erfolgen.

Literaturverzeichnis

[1] Saake, Heuer, Sattler; Datenbanken: Implementierungstechniken 2.Auflage; Bonn; mitp-Verlag; 2005

[2] Wikipedia: Datensicherung (Stand: 17.07.2006)
Internet: http://de.wikipedia.org/wiki/Datensicherung

[3] Schumacher, Stefan: Methoden zur Datensicherung (Stand: 23.05.2006)
Internet: http://net-tex.dnsalias.org/~stefan/backup.pdf

[4] Warum MySql? (Stand: 15.07.2006)
Internet: http://www.mysql.de/why-mysql/

[5] MySQL 5.1 Referenzhandbuch (Stand:20.07.2006)
Internet: http://dev.mysql.com/doc/refman/5.1/de/index.html

Abkürzungsverzeichnis

Abkürzung	Bedeutung
API	application programming interface (Schnittstelle zur Anwendungsprogrammierung)
DBMS	Datenbank Management System
FAT	File Allocation Table (Dateisystem von DOS und Windows)
JDBC	Java Database Connectivity (Standartschnittstelle von Java für die Verbindung zu Datenbanken)
LAMP	Linux Apache MySql PHP
NTFS	New Technology File System (Dateisystem von Windows NT)
ODBC	Open DataBase Connectivity (Standartschnittstelle für die Verbindung zu Datenbanken)
PHP	PHP: Hypertext Preprocessor (ursprünglich: Personal Home Page Tools)
WAMP	Windows Apache MySql PHP

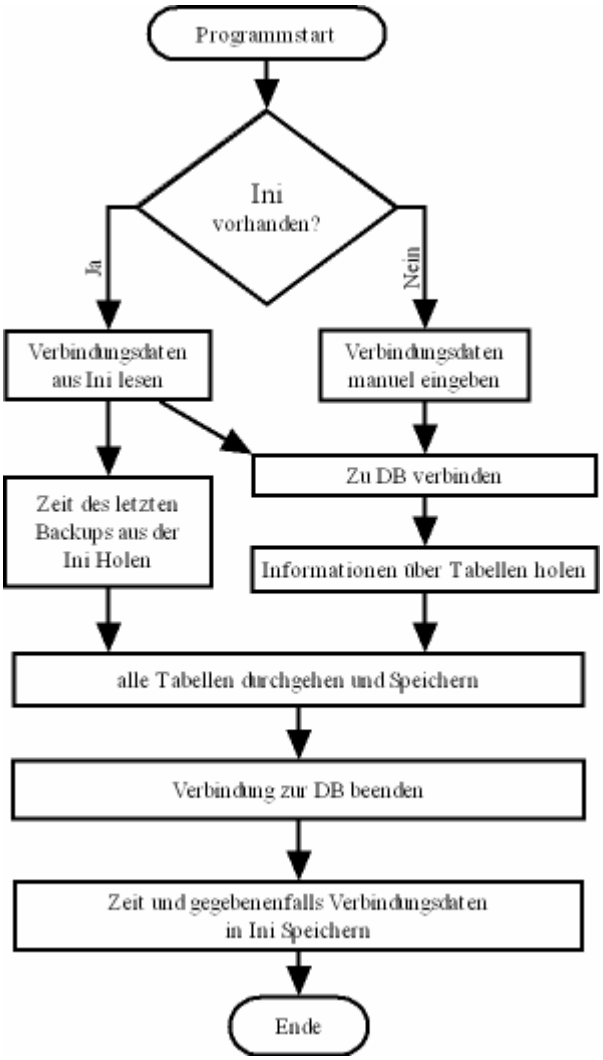
Tabellen- und Abbildungsverzeichnis

Anlagen

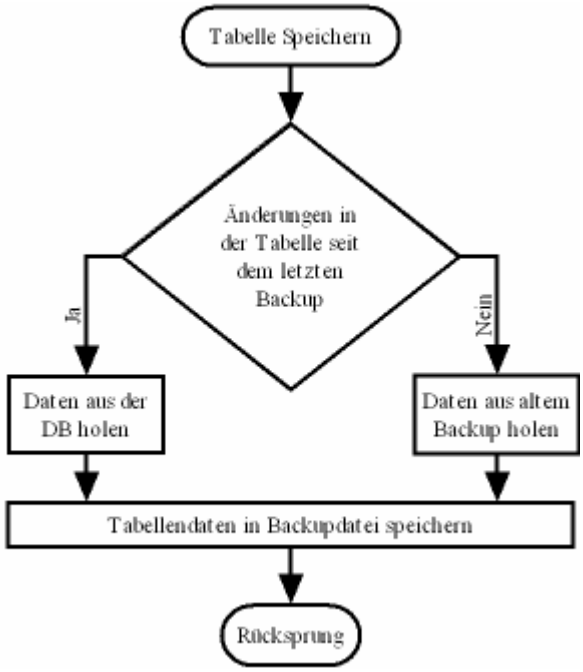
Das in Kapitel 5.2 besprochene Programm (FastMySqlBackup) ist mit Quelltext auf der beiliegenden CD zu finden.

Auf meiner Homepage unter: http://www.mcs-soft.de/s_11.html findet sich immer die aktuellste Version des Programms und der dazugehörige Quelltext.

Grober Programmablaufplan des Programms FastMySqlBackup



Grober Programmablauf der Funktion „Tabelle speichern“



Selbständigkeitserklärung

Ich versichere, dass ich die vorliegende Praxisarbeit ohne fremde Hilfe selbständig verfasst und nur die angegebenen Quellen und Hilfsmittel benutzt habe. Wörtlich oder dem Sinn nach aus anderen Werken entnommene Stellen sind unter Angabe der Quellen kenntlich gemacht. Die Arbeit wurde bisher in gleicher oder ähnlicher Form weder veröffentlicht, noch einer anderen Prüfungsbehörde vorgelegt.

Leipzig, 21.07.2006 _____

Christian Gräfe