Universität Rostock

Fakultät für Informatik und Elektrotechnik

Masterarbeit

zur Erlangung des akademischen Grades
Master of Science

Thema:	Konzeption der Datenintegration für eine zu entwickelnde Benthos-Datenbank
Autor:	B.Sc. Jessica Zierke MatNr. 8200700
Version vom:	28. April 2014
1. Gutachter:	Prof. Dr. Andreas Heuer
2. Gutachter:	Dr. Michael Zettler
Betreuer:	Ilvio Bruder, Susanne Jürgensmann, Dr. Susanne Feistel

Abstract

Das Leibniz-Institut für Ostseeforschung Warnemünde erhebt in der Arbeitsgruppe „Ökologie benthischer Organismen“ Daten über das Vorkommen von Organismen in der Bodenzone eines Gewässers. Die erfassten Daten werden derzeit in Excel-Tabellen protokolliert. Die vorliegende Arbeit befasst sich mit der Konzeption eines Datenintegrationsprozesse, um sämtliche in Excel gespeicherten Daten in eine Datenbank zu transformieren. Hierzu werden die Daten über die modellierte Datenbank in einer einheitlichen Struktur repräsentiert. Die vorliegende Arbeit bildet die Grundlage für die Integration der in Excel gespeicherten Daten. In diesem Zusammenhang wird ein Konzept vorgestellt, um die zu integrierenden Daten in den Excel-Tabellen zu adressieren und gleichzeitig in die Datenbank zu transformieren.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungsverzeichnis

CSV Comma-separated values

DBMS Datenbankmanagementsystem

ER Entity-Relationship

ETL Extraktion-Transformation-Laden

ELT Extraktion-Laden-Transformation

IOW Leibniz-Institut für Ostseeforschung Warnemünde

SQL Structured Query Language

VBA Visual Basic for Application

XML Extensible Markup Language

1. Einführung

Das Leibniz-Institut für Ostseeforschung Warnemünde (IOW) befasst sich mit der interdisziplinären Meeresforschung in Küsten- und Randmeeren. Dazu zählen die vier Wissenschaftsbereiche Physikalische Ozeanographie, Meereschemie, Biologische Meereskunde sowie Marine Geologie. Dabei steht die Erforschung des Ökosystems der Ostsee im Vordergrund [6].

1.1. Motivation

In der Arbeitsgruppe „Ökologie benthischer Organismen“ erhebt das IOW Daten über das Vorkommen von Benthos-Arten, speziell das tierische Benthos, und vorhandenen Umweltbedingungen. In [10] wird der Begriff Benthos wie folgt definiert:

> *„Benthos bezeichnet die Gesamtheit aller Lebewesen auf dem Meeresboden und in allen Tiefenzonen.“*

Es sind bestimmte Verfahren entwickelt worden, um benthische Organismen, sowohl ober- als auch unterhalb der Sedimentoberfläche, zu beproben [13].
Im Laufe der Jahre hat sich eine große Menge an Informationen angesammelt. Die erfassten Daten, die unter anderem Fundort, Zeitpunkt, Arten und Individuenzahlen enthalten, werden derzeit manuell in Excel-Eingabeprotokolle eingetragen. Der gesamte Datensatz erstreckt sich über einen Zeitraum von mehr als 120 Jahren und stammt aus eigenen Untersuchungen der Arbeitsgruppe „Ökologie benthischer Organismen“ und aus Literaturrecherchen. In den Datensätzen werden zusätzlich Informationen über die Organismen sowie jeweils pro Fund Metadaten wie Ort, Zeit und vorhandene Umweltbedingungen gespeichert. Für den Ostseeraum lassen sich die Lebewesen etwa 2000 Arten zuordnen, die in einer sich ändernden Taxonomie erfasst sind.
Die Excel-Eingabeprotokolle wurden den wachsenden wissenschaftlichen Ansprüchen angepasst und weisen deutliche Strukturunterschiede auf. Die Ergebnisse werden separat in Excel-Ausgabeprotokollen und zusammenfassend als Mittelwerte gespeichert.
Aufgrund der einfachen Anwendung und Verwaltung ist das Tabellenkalkulationsprogramm Excel für die Eingabe der Datensätze gewählt worden. Excel bietet die Möglichkeit Datensätze in kurzer Zeit auszuwerten und unter anderem für weitere Anwendungen zur Verfügung zu stellen. Weitere Vorteile von Excel bezüglich der Verarbeitung und Auswertung der Eingabedaten liegen in mathematischen Rechenoperationen. Ergebnisse können effektiv über komplexe Formeln berechnet und unter anderem in Ausgabeprotokollen veranschaulicht werden.
Für die derzeitige Dateneingabe bietet Excel einen guten Lösungsansatz. Die Verwaltung der Datensätze mit Excel ist jedoch aufwändig und für eine langfristige Speicherung und Auswertung der großen Menge an Daten nicht geeignet. Excel stößt bezüglich der Verwendung als Langzeitdatenspeicher und Datenverwaltung an Grenzen. Hierbei besteht die Gefahr, dass über die Jahre erfasste Daten durch einen unerwünschten Effekt gelöscht werden können. Weiterhin ist mit der Anwendung von Excel kein Mehrbenutzerzugriff möglich, dass heißt mehrere Anwender können nicht zur gleichen Zeit auf die Datensätze zugreifen.

Außerdem führt das steigende Volumen der protokollierten Daten zusätzlich zu hohen Zugriffszeiten in Excel.
Der Lösungsansatz über Excel wird zunehmend unflexibel, denn die Software stellt keine Datenbank, sondern ein Tabellenkalkulationsprogramm dar [2]. Die Verwaltung bedarf einer Vereinfachung, unter anderem bezüglich der Speicherung, Eingabe und Auswertung der Datensätze. Die genannten Nachteile in Bezug auf die Problemstellung machen die Nutzung einer Datenbank notwendig. Die Datenbank hat die Aufgabe, unter anderem die gesamten Datensätze, die bisher in Excel verwaltet werden, zu speichern sowie komplexe Auswertungen zu ermöglichen.
Eine Datenbank bietet eine dauerhafte, zentrale, widerspruchsfreie und effiziente Speicherung von großen Datenmengen. Die wesentlichen Merkmale eines Datenbankmanagementsystem (DBMS) bestehen in der dauerhaften Organisation langfristig zu haltender Daten. Die Daten müssen etliche Anwendungen, wie Modifikation, Selektion oder Operationen, überdauern. Zugriffskontrolle und Datensicherheit wird automatisch durch ein Datenbanksystem gewährleistet, sodass kein unbefugter Zugriff sowie ungewollter Datenverlust der vertraulichen Daten entsteht. Dies kann unter anderem durch das Festlegen von Benutzerrollen realisiert werden. Durch die Unterstützung des Transaktionskonzeptes[1] wird sichergestellt, dass sich die Daten zu jeder Zeit in einem konsistenten Zustand befinden. Viele Benutzer können parallel mit den Datenbeständen arbeiten, ohne sich gegenseitig zu beeinflussen und damit unerwünschte Nebeneffekte wie Löschen oder Überschreiben der Daten, zu vermeiden [29].
Für die Speicherung der bisher in Excel verwalteten Datenbestände eignet sich eine Datenbank. Diese ermöglicht die komplette Verwaltung sowie Funktionen wie Abrufen, Sortieren, Analysieren und Zusammenfassen der Datensätze. Für die neu zu erstellende Benthos-Datenbank ergeben sich entsprechende Vorteile. Aus den einzelnen Eingabedaten können umfangreiche Ausgabeprotokolle generiert und ausgegeben werden. Dies ist das große Ziel des Umstieges auf eine Datenbank und ermöglicht eine geordnete, präzise und effiziente Dateneingabe.

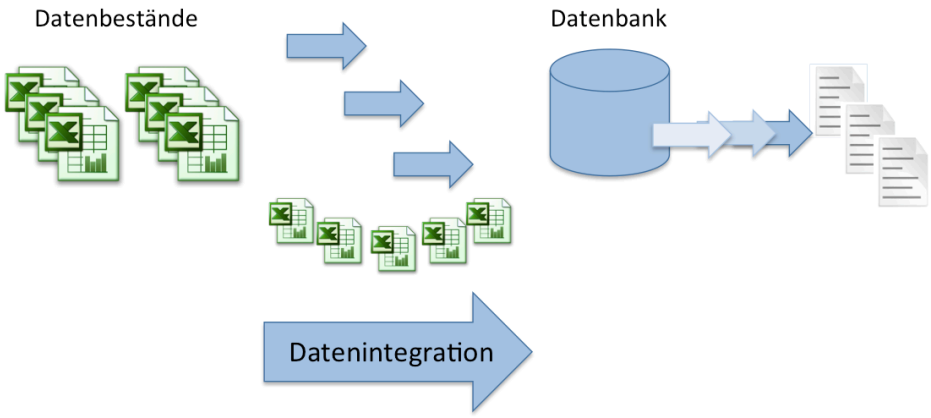


Abbildung 1: Integration der Datenbestände in eine Datenbank

[1] Über das Transaktionskonzept wird die Datenbank von einem konsistenten Zustand in einen anderen ebenfalls konsistenten Zustand überführt

Mit der Zeit haben sich etliche Datenbestände angesammelt. Die vorhandenen Datenbestände müssen in die Datenbank integriert werden. Dieser Prozess ist in der Abbildung 1 grafisch dargestellt und wird unter dem Begriff *Datenintegration* zusammen gefasst.
Die Abbildung 1 symbolisiert auf der linken Seite die zu integrierenden Informationen, die in umfangreichen Excel-Dokumente gespeichert sind und auf der rechten Seite die zu entwickelnde Datenbank, welche die Datenbestände aus den Excel-Dokumenten vereinheitlicht speichern soll. Aus den gespeicherten Daten soll die spätere Generierung von Ausgabeprotokollen möglich sein.

1.2. Zielstellung

Das Ziel der vorliegenden Arbeit ist die Konzeption eines Datenintegrationsprozesses für eine zu entwickelnde Benthos-Datenbank. Die Zielstellung der vorliegenden Arbeit wird in Abbildung 2 symbolisiert.

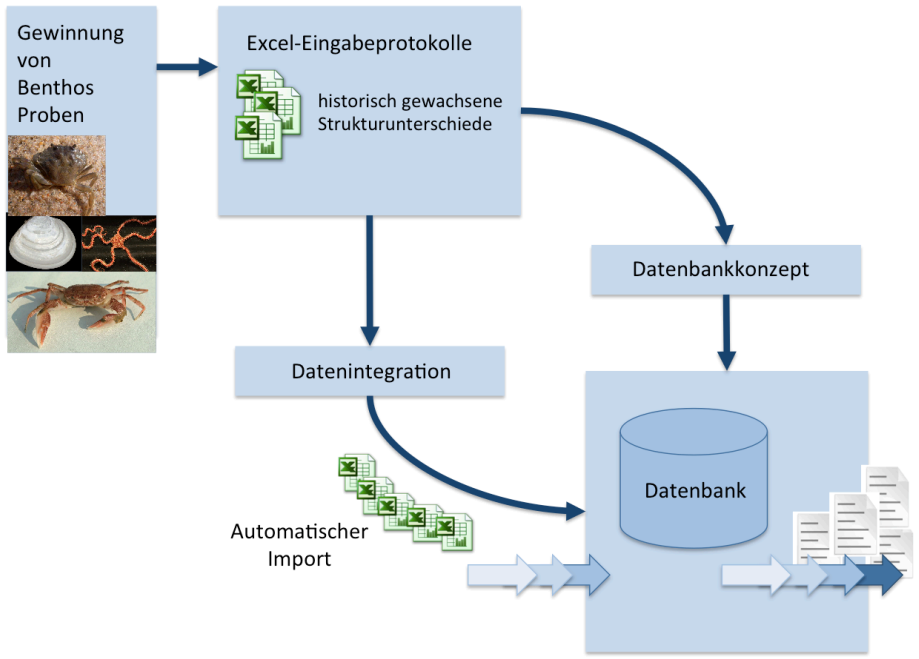


Abbildung 2: Zielstellung

Im ersten Schritt ist es notwendig eine inhaltliche und strukturelle Analyse der gesamten Excel-Eingabeprotokolle vorzunehmen, um daraus ein Datenbankkonzept ableiten zu können. Es werden verschiedene Schritte durchlaufen, mit deren Hilfe ein Datenbankkonzept in ein geeignetes Datenbankmodell umgesetzt werden kann. Anforderungen an das neue System müssen identifiziert, abstrahiert sowie im Modell und in der Datenbank umgesetzt werden.
Der Schwerpunkt der Arbeit ist die Integration der vorliegenden Datensätze. Die Quellen für die Datenintegration sind detaillierte Excel-Eingabeprotokolle, die deutliche Unterschiede in der Struktur aufweisen und daher als unterschiedliche Excel-Protokolltypen zu betrachten sind. Weiterhin sind pro Fund verdichtete Daten mit Taxonomie-Informationen und Umrech-

nungsfaktoren erfasst worden. Ziel der Datenintegration ist eine MySQL[2]-Datenbanklösung, in der möglichst umfangreich sämtliche in Excel vorliegende Daten zu erfassen sind.
Die einzelnen Prozesse auf MySQL sollen möglichst in PHP[3] umgesetzt werden. Durch die Arbeit soll der spätere automatische Import der Excel-Eingabeprotokolle vorbereitet werden.

1.3. Aufbau der Arbeit

Die vorliegende Arbeit umfasst einschließlich der Einleitung sieben Kapitel.
In Kapitel 2 werden die biologischen Grundlagen vorgestellt. In diesem Kapitel wird zusätzlich auf die Arbeitsgruppe *Ökologie benthischer Organismen* des IOW eingegangen. In diesem Zusammenhang wird die Gewinnung der Benthos-Proben und gleichzeitig die Untersuchung der Proben im Labor beschrieben.
Der aktuelle Forschungsstand wird in Kapitel 3 geschildert. Hierbei werden die Grundlagen sowohl der Datenintegration, als auch der Datenbankmodellierung vorgestellt. In diesem Kapitel werden zusätzlich zwei Ansätze für die Analyse von Excel-Tabellen erläutert. Die in diesem Kapitel vorgestellten Konzepte und Techniken bilden die Basis der Arbeit.
In Kapitel 4 wird eine Analyse der bezüglich der Datenintegration auftretenden Probleme vorgenommen. Hierzu werden in diesem Kapitel sowohl eine inhaltliche-, als auch eine strukturelle Analyse der Excel-Eingabeprotokolle durchgeführt.
Auf Basis der vorherigen Kapitel wird in Kapitel 5 ein Konzept entwickelt, welches die Integration möglichst aller in Excel gespeicherten Daten ermöglichen soll. Hierzu ist das Kapitel in zwei Abschnitte unterteilt. Im ersten Abschnitt wird das Datenbankkonzept für die Modellierung eines einheitlichen Zielschemas vorgestellt. Im zweiten Abschnitt wird die Konzeption der Datenintegrationsprozess beschrieben. Zum einen werden in diesem Zusammenhang verschiedene Ansätze für die Adressierung der Daten in Excel und zum anderen das Schema Mapping[4] erläutert. Hierzu wird die Überführung der Daten aus den Excel-Eingabeprotokollen in das Zielschema grafisch veranschaulicht.
Die Implementierung wird in Kapitel 6 beschrieben. Hierbei wird die Umsetzung anhand der vorgestellten Ansätze des Konzeptes vorgenommen.
Das abschließende Kapitel 7 In dem abschließenden Kapitel 7 werden die Ergebnisse der Arbeit zusammengefasst und ein Ausblick auf Themen für mögliche Folgearbeiten gegeben.

[2] MySQL ist ein relationales Datenbankverwaltungssystem, welches als Open-Source-Software verfügbar ist.
[3] PHP ist eine Skriptsprache für das Erstellen dynamischer Webseiten oder Webanwendungen.
[4] Das Schema Mapping setzt äquivalente Bestandteile zweier unterschiedlicher Schemata miteinander in Verbindung.

2. Biologische Grundlagen

Der Meeresboden, auch *Benthal* genannt, beherbergt eine vielfältige Flora[5] und Fauna[6], die unter dem Begriff *Benthos* zusammengefasst werden. Seit dem 19. Jahrhundert führen Wissenschaftler Untersuchungen durch, um Informationen über die am Boden der Ostsee lebenden Organismen zu erhalten [27, S. 161]. Diese Informationen werden u.a. genutzt, um die Lebensraumzustände und Umweltbedingungen zu erfassen. Dazu sind Erfassungsprogramme, wie Inventarisierung und Monitoring, notwendig.
In diesem Kapitel werden die biologischen Grundlagen erläutert. In Abschnitt 2.2 wird die Gewinnung von Benthosproben in der AG „Ökologie benthischer Organismen“[7] beschrieben.

2.1. Benthos

Die bodenlebenden Organismen sind an der Schnittstelle zwischen Meeresboden und Wasserkörper (siehe Abbildung 3) zu finden. Um die benthischen Organismen vom Plankton[8] und vom Nekton[9] des Freiwasserbereichs abgrenzen zu können, wurde der Begriff 1890 von Ernst Haeckel[10] eingeführt. Dabei wird zwischen Zoobenthos[11] und Phytobenthos[12] unterschieden.

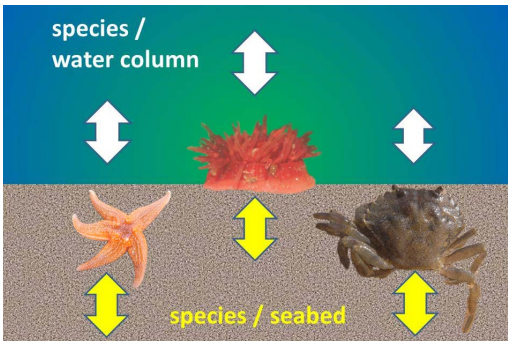


Abbildung 3: Schnittstelle zwischen Wasserkörper und Meeresboden [10]

Die benthischen Organismen können unter anderem nach ihrer Größe eingeteilt werden. Hierbei wird zwischen Makro-, Meio- und Mikrozoobenthos unterschieden. Die Makrozoobenthosarten sind mit mehr als 1mm die Größten der benthischen Organismen. Insgesamt sind mehr als 2000 unterschiedliche Arten in der Ostsee bekannt [16].
Das Leben im Meeresboden bietet mehr Schutz vor Gefahren wie Räubern. So bevorzugen Krebse, Würmer und andere Arten das Leben in der Bodenzone eines Gewässers [13]. In der Abbildung 4(a) ist eine *Wellhornschnecke* und in der Abbildung 4(b) eine *Strandkrabbe* abgebildet. Die Benthos-Organismen sind an spezielle Standortbedingungen angepasst. Das

[5] Flora bezeichnet die Gesamtheit der Pflanzenarten.
[6] Fauna bezeichnet die Gesamtheit der Tierarten.
[7] Arbeitsgruppe des Leibniz-Institut für Ostseeforschung Warnemünde
[8] Organismen, die im Wasser leben, Schwimmrichtung wird durch Strömungsrichtung beeinflusst
[9] Gesamtheit der Tiere in Ozeanen/Binnengewässern, Schwimmrichtung ist strömungsunabhängig
[10] Deutscher Zoologe, Philosoph und Freidenker, starb 1919 in Jena
[11] Tierisches Benthos
[12] Pflanzliches Benthos

führt dazu, dass sich spezielle Gemeinschaften bilden, die an unterschiedlichen Lebensräumen wie Hartböden, Riffen, Sandböden, Schlamm und Schlick vorkommen. Dabei spielt die Gesamtheit der Faktoren unter anderem aus Sedimentzusammensetzung, Nahrungsverfügbarkeit sowie hydrografischen Faktoren eine entscheidende Rolle [10, 12, 15].

(a) Wellhornschnecke (b) Strandkrabbe

Abbildung 4: Benthos-Arten [16]

2.2. Gewinnung von Benthosproben

Das IOW erhebt in der AG „Ökologie benthischer Organismen“ Daten über das Vorkommen von benthischen Organismen und vorhandenen Umweltbedingungen.
Das „Baltic Monitoring“ Programm der HELCOM[13] zählt unter anderem zu einer der wichtigsten Aufgaben des IOW. Bei dem Programm werden mit einem Forschungsschiff acht Stationen zwischen *Kieler Bucht* und *Usedom* angefahren, an denen die am Boden lebenden Organismen untersucht werden sollen. An den jeweils angefahren Station werden unter anderem Daten wie Position, Temperatur, Wassertiefe, Salz- sowie Sauerstoffgehalt erfasst.

Die nachfolgende Abbildung 5 zeigt eine vereinfachte Darstellung der aufwändigen Erhebung der Organismen im Meeresboden.

[13] Zwischenstaatliche Kommission, die für den Schutz der Meeresumwelt im Ostseeraum arbeitet.

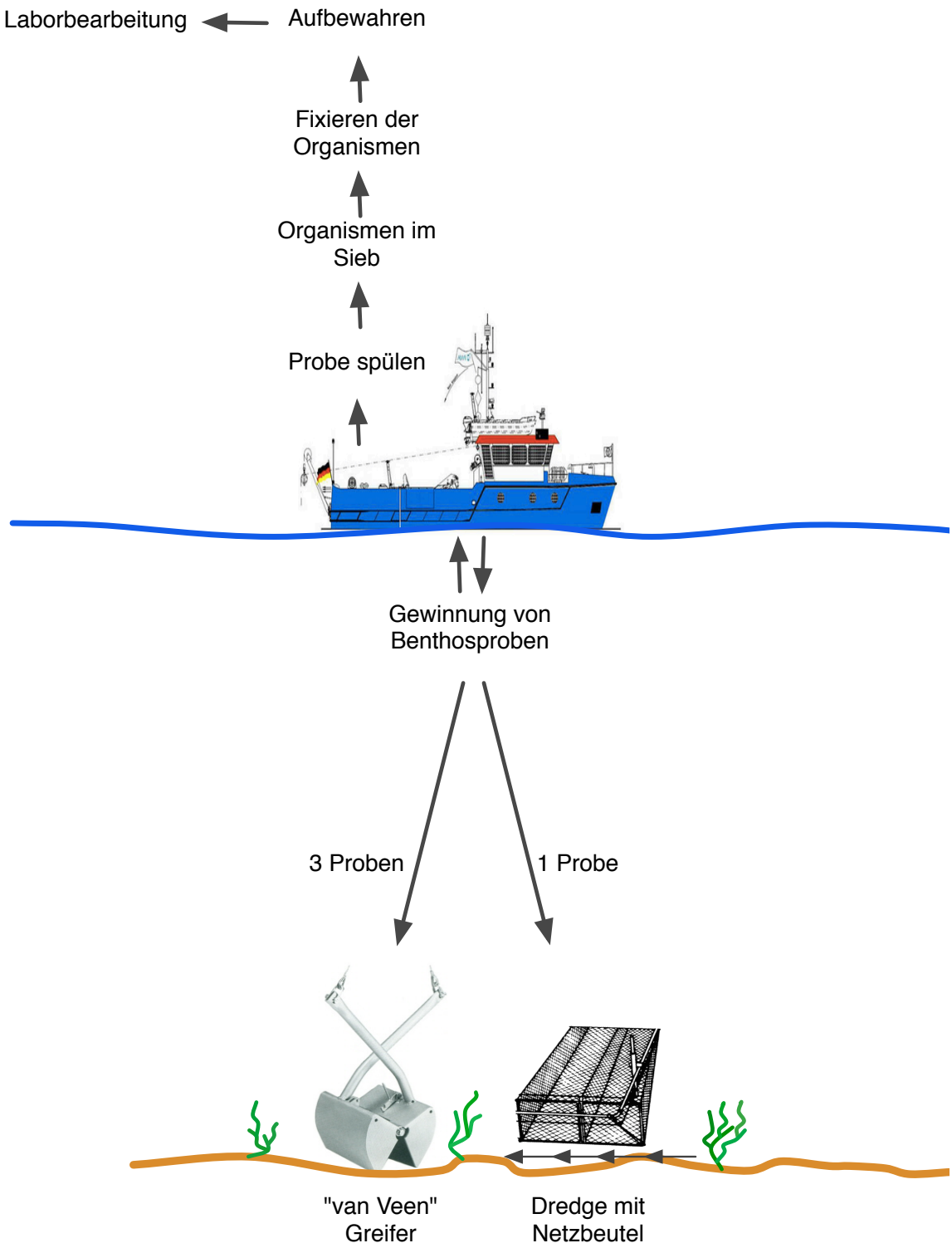
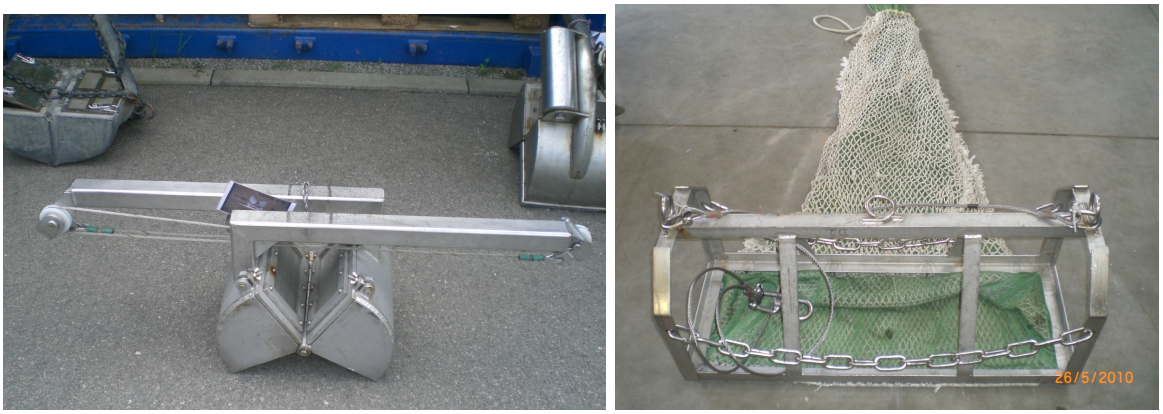


Abbildung 5: Gewinnung von Benthosproben

Für die Probenahme von bodenlebenden Organismen existieren verschiedene Geräte, unter anderem der *„van Veen“ Greifer* (Abbildung 6(a)) sowie die *Dredge* (Abbildung 6(b)). Mit Hilfe des Greifers können die Organismen im Meeresboden und mit der Dredge können schnellflüchtige Organismen auf dem Meeresboden erfasst werden [17]. Es existieren allerdings noch zusätzliche Geräte. Jede Art von Probenahme wird als *Hol* bezeichnet.
Der Greifer wird vom Schiff aus über einen Kran ins Wasser gelassen und auf dem Meeresboden abgesetzt. Dabei drückt sich der Greifer in die Sedimentoberfläche und wird mit Sediment gefüllt. Anschließend schließt sich der Greifer und wird wieder an Bord geholt. Der Greifer hat eine Ausstichfläche von rund 0,1m^2. Es werden jeweils an einer Station drei Proben mit dem Greifer entnommen.
Mit der Dredge können Organismen der oberen Sedimentschicht erfasst werden. Diese wird wie der Greifer mit einem Kran bis zum Meeresboden ins Wasser gelassen und eine gewisse Zeit auf der Sedimentoberfläche geschleppt. Die Zeit ist von der Beschaffenheit des Sedimentes abhängig. Während des Schleppens werden die Organismen in einem Netzbeutel aufgefangen und wieder an Bord geholt. Es wird jeweils an einer Station eine Probe mit der Dredge entnommen.
An Bord werden die Proben für eine spätere Laboruntersuchung vorbereitet (siehe dazu Abbildung 5). Dabei werden die Proben gründlich gewaschen, um das Sediment heraus zu spülen. Die sauberen Organismen der Greiferproben befinden sich in einem Sieb mit einer Maschenweite von 1mm, die sauberen Organismen der Dredge hingegen befinden sich in einer Auffangwanne. Anschließend werden die sauberen Organismen der Proben in spezielle Gefäße gefüllt, fixiert und für die spätere Laboruntersuchung aufbewahrt [17].



(a) Greifer „van Veen“ (b) Dredge „Kieler Kinderwagen“

Abbildung 6: Konventionelle Sammelverfahren [17]

Um die Bearbeitung der Proben im Labor besser verstehen zu können, werden zunächst einige Begrifflichkeiten definiert.

Taxon, ist eine Gruppe von Lebewesen, die aufgrund fehlender oder nicht erkennbarer artspezifischer Merkmale nicht der fachlichen Definition einer Art entsprechen muss. Es existieren verschiedene taxonomische Stufen, unter anderem Art, Gattung, Familie [22].

Taxa, ist die Mehrzahl von Taxon.

Makrozoobenthos, ist die Fauna des Meeresbodens, die auf oder teilweise bzw. vollständig im Boden lebt und von einem Sieb mit 1mm Maschenweite zurückgehalten wird [22].

Abundanz, Ist die Anzahl der Individuen auf eine bestimmte Fläche (z.B. $1m^2$) bezogen [22].

Biomasse, ist die Menge der biogenen Substanz in Form der Feucht-, Trocken- bzw. der Aschefreien Trockenmasse zu einem bestimmten Zeitpunkt und wird ebenfalls auf eine Bezugsfläche (z.B. $1m^2$) hochgerechnet [22].

FM, Feuchtmasse.

TM, Trockenmasse.

AFTM, Aschefreie Trockenmasse.

Tara, ist die Leermasse eines Wiegeschälchens [22].

Probenschale, auch Hauptschale genannt. Die Schalen werden mit dem Stationskürzel, der Holnummer, dem Probenahmedatum und dem Name des Bearbeiters gekennzeichnet [22].

Hol, ist die Verwendung eines Probenahmegerätes, z.B Greifer oder Dredge, auf dem Schiff.

Nach Erhebung der Proben gelangen die fixierten Benthosproben für eine Untersuchung in das Labor. Die zu untersuchenden Proben werden zunächst über ein Laborsieb gespült und in Probeschalen überführt. Anschließend erfolgt das Aussortieren. Hierbei werden die Individuen nach ihrer Art oder Großtaxa ausgesammelt und in Sortiergefäßen gelagert. Nach der Aussortierung erfolgt die Bestimmung der Abundanz der einzelnen Taxa auf Grundlage der aktuellsten Artenliste. Dabei werden die unterschiedlichen Taxa mit einer Zähluhr gezählt. Nachdem sämtliche Taxa gezählt wurden, müssen diese für eine exakte Biomassenbestimmung gewogen werden. Die Biomasse des Makrozoobenthos, die aus den Greifern gewonnenen Organismen, kann als FM, TM und/oder AFTM ermittelt werden.

1. **FM**: Als erstes muss das Tara-Gewicht des Schälchens bestimmt werden. Anschließend müssen die Organismen vom überschüssigen Wasser befreit werden. Die Organismen werden hierzu auf saugfähiges Papier gelegt und wieder in das Wiegeschälchen gegeben. Das Wiegeschälchen mit den enthaltenen Organismen wird anschließend auf eine Waage gestellt. Um das Gewicht der FM zu erhalten, muss das Tara-Gewicht vom angezeigten Gewicht auf der Waage abgezogen werden [22].

2. **TM**: Wenn die FM $<$ 100g ist, dann werden die Organismen 15 Stunden bei 60 Grad in einem Ofen getrocknet. Wenn die FM $>$ 100g ist, dann müssen die Organismen mindestens 24 Stunden getrocknet werden. Nach der Trocknung müssen die Organismen abkühlen und werden anschließend gewogen. Um das Gewicht der TM zu erhalten, muss das Tara-Gewicht vom angezeigten Gewicht abgezogen werden [22].

3. **AFTM**: Wenn die FM $<$ 100g ist, dann werden die Organismen 5 Stunden bei 500 Grad in einem Muffelofen verascht. Wenn die FM $>$ 100g ist, dann müssen die Organismen 10 Stunden verascht werden. Nach der Veraschung werden die Proben gewogen. In dem Schälchen befindet sich nur noch die Asche. Um das Gewicht der AFTM zu erhalten, muss das Tara-Gewicht vom angezeigten Gewicht abgezogen werden [22]. Der organische Anteil des Tiermaterials wurde verbrannt, der Rest ist anorganisch (die Asche). Da jedoch der reale organische Anteil der Organismen benötigt wird, muss das Gewicht der Asche von der TM abgezogen werden.

Die erfassten Ergebnisse werden zusammenfassend in Protokollen gespeichert. Das Verfahren zur Bestimmung der Biomasse, das Durchwiegen, ist aufwändig. Um den Prozessaufwand zu minimieren, wurden über Jahre Umrechnungsfaktoren bestimmt, um die Gewichte ineinander umrechnen zu können. Das heißt, in der Mehrzahl der Biomassenbestimmung ist kein Durchwiegen mehr notwendig. Es wird lediglich die FM gewogen. Aus der FM und den Umrechnungsfaktoren können dann die TM und die AFTM berechnet werden.

Mit der Zeit haben sich etliche Daten auf Sample- und auf Stationsebene angesammelt. Auf der Sampleebene, der sogenannten Greiferebene, handelt es sich um Werte, die sich auf ein bestimmtes Probenahmegerät, z.B. einem Greifer, beziehen. Diese Geräte können verschiedene Ausstichflächen aufweisen. Die Ausstichfläche muss jeweils protokolliert werden. Aus den Replikaten, den Parallelen an einer Station, werden Mittelwerte berechnet und auf $1m^2$ hochgerechnet und anschließend als Mittelwerte in einer weiteren Excel-Tabelle festgehalten. Diese Daten befinden sich derzeit auf Stationsebene, die zum Teil aus den eigenen Untersuchungen und aus Literaturrecherchen stammen. In der Regel werden je Station drei Parallelen mit dem Greifer genommen.
Derzeit werden die Daten sowohl auf Sample-, als auch auf Stationsebene in Eingabeprotokolle gespeichert. Ziel ist die Transformation sämtlicher Datensätze, die sich auf Sampleebene befinden, in das neue System zu überführen und die Daten über eine einheitliche Darstellung zu repräsentieren. Hierzu ist eine Erfassung jedes Excel-Eingabeprotokolls notwendig, um die Datensätze möglichst vollständig zu integrieren und die Daten jedes Samples bzw. Hols zu speichern.
Zusätzlich sollen auch die Informationen, die als Stationsmittelwerte (Stationsebene) gespeichert sind, übernommen werden. Diese Werte sollen, wenn möglich, nicht neu in das System eingetragen werden, sondern automatisch integriert werden.
Weiterhin soll die Möglichkeit geschaffen werden, die Daten, die aus Literaturrecherchen stammen, ebenfalls im neuen System speichern und eingeben zu können.
Für die erfassten Benthos-Arten soll ein automatischer Taxonomie-Check ermöglicht werden, um die Taxonomie-Informationen abzufragen. Der Check ist notwendig, um die Taxonomie regelmäßig in der Datenbank zu aktualisieren.
Das Ergebnis der zu modellierenden Datenbank und des Datenintegrationsprozesses ist die Speicherung der Daten sowohl auf Sample-, als auch auf Stationsebene. Gleichzeitig soll die Möglichkeit bestehen, die Daten von der einen Ebene in die andere Ebene umrechnen zu können, da die Daten der Sampleebene benötigt werden, um unter anderem Statistiken aufstellen zu können.

3. Stand der Forschung

In diesem Kapitel wird ein Überblick über grundlegende Themen sowohl der Datenintegration, als auch der Datenbankmodellierung angegeben. Die in diesem Kapitel vermittelten Grundlagen dienen als allgemeine Einführung in die Techniken die Bestandteil der vorliegenden Arbeit sind. Zusätzlich werden Themengebiete erläutert, die nicht Bestandteil der vorliegenden Arbeit sind. Die Abbildung 7 zeigt eine schematische Darstellung der zu betrachteten Grundlagen bezüglich der Datenintegration und der Datenbankmodellierung. Da die Datensätze tabellarisch in Excel-Dateien gespeichert sind, werden zusätzlich in diesem Kapitel zwei Ansätze für die Analyse von Excel-Tabellen vorgestellt.

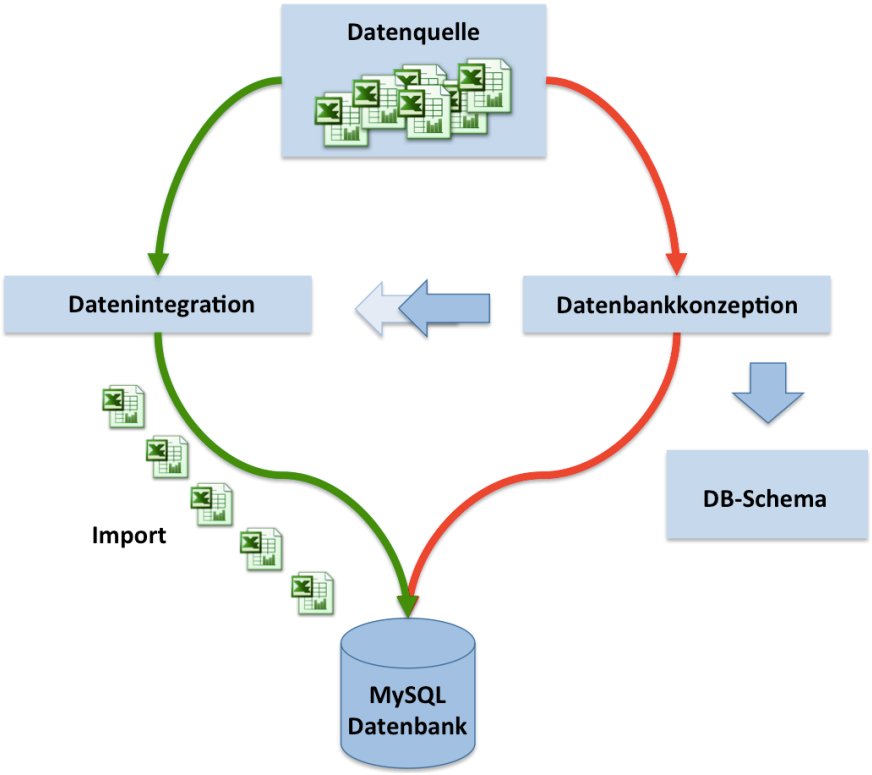


Abbildung 7: Überblick der zu betrachtenden Grundlagen bezüglich Datenintegration und Datenbankmodellierung

3.1. Datenintegration

Das Zusammenführen von Informationen aus verteilten, heterogenen[14] Datenquellen stellt nicht nur in der Datenbankforschung, sondern auch in vielen Unternehmen eine große Herausforderung dar. Über den Prozess der Datenintegration wird eine einheitliche und zentrale Speicherung von Datenbeständen angestrebt und dabei eine weitestgehend redundanzfreie Repräsentation der Informationen geschaffen.

[14] Heterogen bedeutet nicht gleichartig.

Felix Naumann definiert Informationsintegration wie folgt:

> *„Informationsintegration ist die korrekte, vollständige und effiziente Zusammenführung von Daten und Inhalt verschiedener, heterogener Quellen zu einer einheitlichen und strukturierten Informationsmenge zur effektiven Interpretation durch Nutzer und Anwendungen.“* [25]

3.1.1. Grundprobleme der Datenintegration

Im weiteren Verlauf werden die drei Grundprobleme der Datenintegration erläutert. Dabei werden die Probleme der Verteilung und Autonomie kurz vorgestellt und anschließend das Problem der Heterogenität detailliert beschrieben. In der vorliegenden Arbeit stellt hauptsächlich die Überwindung Letzteres eine Schwierigkeit dar.

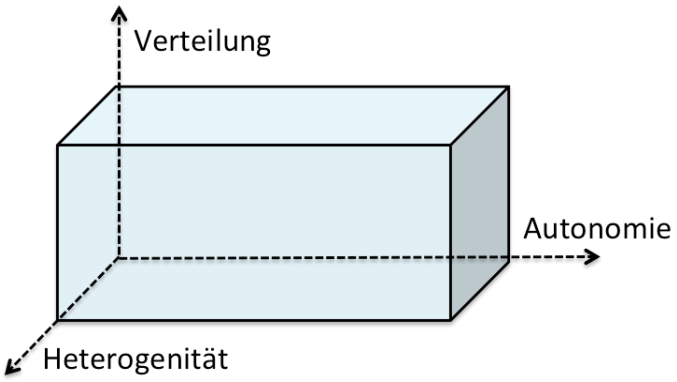


Abbildung 8: Orthogonale Dimensionen in der Datenintegration [24]

Die drei Grundprobleme in der Datenintegration bilden die *orthogonalen Dimensionen* (siehe Abbildung 8). In jeder der drei Dimensionen ist es durchaus möglich, dass die Probleme unabhängig voneinander vorkommen können [24]. In der Regel treten alle drei Probleme gemeinsam auf:

> *„...das heißt, Datenquellen sind verteilt und heterogen und werden von autonomen Organisationen betreut“[24]*

Verteilung

Ein Problem der Integration sind Datenbestände, die in verteilten Informationssystemen vorliegen. Man unterscheidet zwei Aspekte der Verteilung, zum einen die *logische* und zum anderen die *pysische* Verteilung.
Bei der Verteilung aus logischer Sicht sind die Daten an verschiedenen Stellen, wie z.B. Schema, Tabellen oder Attribute zu finden [24].
Die physischen Verteilung bezeichnet die geografischen Verteilung von Datenbeständen. Die Daten sind getrennt und teilen sich keine gemeinsamen Komponenten wie z.B. CPU [24, 28, 8].

Autonomie

Autonomie beschreibt, bezugnehmend auf Komponenten wie Aufbau, Zugriff oder Verwaltung, die Unabhängigkeit bestehender Datenquellen untereinander [24]. Hierbei werden vier Arten unterschieden [28], die im Folgenden definiert werden.
Designautonomie ist vorhanden, wenn eine Datenquelle eigenständig entscheiden kann, in welcher Art und Weise die Daten für Andere zur Verfügung gestellt werden. In diesem Zusammenhang wird unter anderem über das jeweilige Datenformat, das verwendete Datenmodell oder die syntaktische Darstellung entschieden. *Schnittstellenautonomie* betrifft die Unabhängigkeit der Entscheidung bezüglich technischer Verfahren, die auf bestimmte Daten zugreifen, hierzu zählen unter anderem eingesetzte Protokolle und Anfragesprachen. *Zugriffsautonomie* betrachtet die Entscheidungsfreiheit bezüglich des Zugriffes auf Datensätze. Aspekte wie Authentifizierung und Autorisierung sowie Lese- und Schreibrechte werden definiert [24, 8].

Heterogenität

Das Hauptproblem in der Zusammenführung von Informationen stellen Systeme dar, die weder gleiche Strukturen noch gleiche Methoden und Modelle für den Zugriff auf die Daten aufweisen [24]. Solche Systeme werden als *heterogene* Informationssystem definiert [28].
Eine der Hauptaufgaben in der Datenintegration ist die Überwindung der Heterogenität. Im weiteren Verlauf des Abschnittes werden die unterschiedlichen Arten der Heterogenität (Abbildung 9) beschrieben.

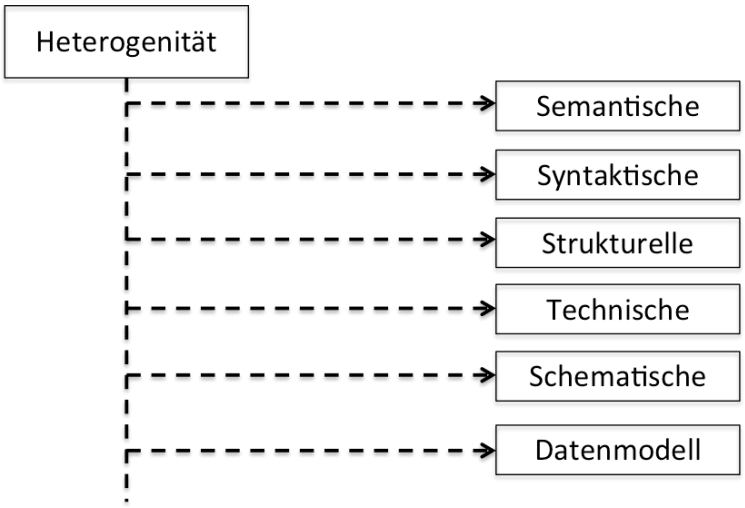


Abbildung 9: Arten von Heterogenität

Technische Heterogenität
Hierbei werden nicht die Unterschiede bezugnehmend auf die Daten sowie deren Repräsentation spezifiziert. Ausschlaggebend bei der *technischen* Heterogenität ist die Realisierung des Zugriffes auf Daten sowie die Verwendung von Anfragesprache, Austauschformat, Kommu-

nikationsprotokolle etc. Es existieren eine Reihe an Methoden, um die technische Heterogenität zu überwinden. Diese Techniken werden in der vorliegenden Arbeit nicht erläutert [24].

Heterogenität auf Datenmodellebene

Für diese Art der Heterogenität ist die Art und Weise der Darstellung von Daten, bezüglich des verwendeten Datenmodells, ausschlaggebend. Strukturierte Systeme haben die Aufgabe Daten durch eine formale Beschreibung ihrer Struktur, das Schema, in einem definierten Datenmodell zu verwalten. Das Datenmodell ist für die Präsentation der Daten zuständig. In den meisten Fällen kommt es vor, dass das Modell für die Modellierung ein anderes Modell ist, als das für die Verwaltung der Daten. Ein Beispiel ist die Modellierung eines relationalen Datenbanksystems. Vorab wird ein ER-Modell definiert, welches die Anforderungen an die Anwendung beschreiben soll. Anschließend wird das ER-Modell in ein relationales Modell umgesetzt. Werden gleiche Daten durch verschiedene Modelle repräsentiert und verwaltet, so liegt Datenmodellheterogenität vor. Die Unterschiede in den verwendeten Datenmodellen haben Auswirkungen auf die Struktur der vorhandenen Schemata [24, 30].

Schematische Heterogenität

Schematische Heterogenität hängt eng mit der strukturellen Heterogenität zusammen und ist ein Spezialfall dieser. Es existieren Ungleichheiten in den verwendeten Komponenten, um die Daten eines gleichen Sachverhaltes zu präsentieren. Bezugnehmend auf das relationale Datenmodell besteht die Möglichkeit, Informationen auf unterschiedliche Art und Weise darzustellen. Zum einen können diese als Relation und zum anderen als Attribut sowie Wert repräsentiert werden. Die Arten können nicht immer klar voneinander getrennt werden und treten meist gemeinsam auf [24].

Strukturelle Heterogenität

Ein Sachverhalt kann mittels eines Schemas auf verschiedene Art und Weise beschrieben werden. Diese Art der Heterogenität liegt vor, wenn Schemata, die gleiche Objekte wieder spiegeln, unterschiedlich sind. Die Unterschiede liegen unter anderem in der Bedeutung und dem eigentlichen Sinn der verwendeten Elemente eines Schemata. Ein gleicher Ausschnitt der realen Welt kann unterschiedlich beschrieben werden. Ursachen der auftretenden Unterschiede sind zum einen verschiedene Anforderungen und zum anderen die Verwendung der Daten sowie technische Beschränkungen. Oft müssen Datenmodelle für bestimmte Anforderungen konzipiert und optimiert werden. Eine weitere Ursache ist die Bestimmungsfreiheit des Modellierers. Die Aufteilung kompletter Namen in Vor- und Nachname ist die Entscheidung des Modellierers und kann daher von Anwendung zu Anwendung unterschiedlich sein. Hierbei ändert sich dann ausschließlich die syntaktische Darstellung des Wertes, nicht die Semantik [24, 19].

Syntaktische Heterogenität

Objekte mit gleicher Semantik können anhand unterschiedlicher Syntax[15] repräsentiert werden. Die Unterschiede liegen unter anderem in Zahlenformaten, Zeichenkodierungen sowie Trennzeichen vor. Ein typisches Beispiel ist die Datumsangabe. So kann zum einen das Datum über das Format YYYY.MM.DD und zum anderen über DD.MM.YYYY dargestellt werden. Es existieren allerdings noch weitere Formate für die Datumsangabe. Trennzeichen können ebenfalls entscheidend sein. Wie bei dem Beispiel des Datums kann das Trennzeichen "-" oder auch "." gesetzt werden. Diese Heterogenität kann bei der Integration gravierende Folgen haben. Es besteht die Möglichkeit, dass Daten verloren oder verfälscht werden können. Die Darstellung von Informationen kann auf unterschiedliche Weise realisiert werden. Die syntaktische Heterogenität wird in [24, S. 65] wie folgt definiert:

> *„Syntaktische Heterogenität in unserem Sinne kann bei der Datenintegration meistens leicht überwunden werden. Zahlenformate lassen sich umrechnen, Zeichendarstellung ineinander transformieren und Trennzeichen durch einfache Parser übersetzen. “*

Es existieren zahlreiche Lösungsansätze. Ein weit verbreitetes Beispiel ist die Längen-, Gewichts- oder auch Zeitangabe. Die Länge kann unter anderem im Zielsystem in Meter, im Quellsystem hingegen in Zentimeter angegeben werden. Diese Werte müssen umgerechnet und im Zielsystem abgelegt werden [24, 19].

Semantische Heterogenität

Bei der semantischen Heterogenität stehen die Probleme der Semantik[16] von Daten, die sich im Quellsystem befinden, im Vordergrund.
Tritt der Fall ein, dass Daten isoliert auftauchen, dann kann für diese Daten keine konkrete Zuordnung, bezüglich des Sachverhaltes, getroffen werden. Das heißt, dass für diese Daten keine Bedeutung zugeordnet werden und somit keine Interpretation dieser erfolgen kann. Dies kann nicht automatisch erfolgen, denn es ist notwendig, diese Daten durch einen menschlichen Nutzer, mit dem notwendigen Hintergrundwissen, auch Kontext genannt, interpretieren zu lassen. So werden meist die Position des Elementes, Kenntnisse über den allgemeinen Anwendungsbereich sowie andere in der Nähe liegende Werte herangezogen, um die Bedeutung eines Wertes genauer bestimmen zu können. Aber auch NULL-Werte sind kritisch zu betrachten und müssen im Zusammenhang mit dem dahinter liegenden Kontext analysiert werden. Dieser Wert kann unter Umständen als *unbekannt*, *nicht erfasst*, oder auch *nicht vorhanden* etc. gelten. Die Beseitigung dieser Konflikte stellt eine Herausforderung in der Zusammenführung von Information in ein Gesamtsystem dar [28, 30, 8].
Bei der Interpretation von Werten ist der Kontext notwendig. Daten können erst durch Berücksichtigung des Kontextes ihre Bedeutung erhalten. So entstehen z.B. Probleme mit Homonymen[17]. Das Wort *Bank* hat zwei Bedeutungen, zum einen die "Sitzbank"' und zum anderen das "Geldinstitut".

[15] Bei der Syntax handelt es sich um die Weise, in welcher die Wörter angeordnet sind, um Bedeutungsbeziehungen der Wörter aufzuzeigen.
[16] Im Gegensatz zur Syntax handelt es sich bei der Semantik um die Bedeutung von Sachverhalten und dem dahinter stehenden Kontext.
[17] Ein Homonym ist ein Wort, welches mehrere Bedeutungen annehmen kann.

Auf die konkrete Bedeutung kann nur implizit geschlossen werden, denn in keinem Datenmodell wird der Kontext angegeben [24].
Das Auffinden von semantischen Konflikten und das Lösen der selbigen ist relativ schwierig. Zu erkennen, ob Namen Synonyme oder Homonyme sind, kann nur im Zusammenhang mit dem notwendigen Hintergrundwissen gelöst werden. Meist sind keine ausführlichen Dokumentationen vorhanden, es stehen meist nur das Schema oder einige Beispieldaten zur Verfügung.
Das komplette Beheben solcher Konflikte ist oftmals schwer möglich. Die Probleme müssen erkannt und entsprechend im neuen System abgelegt werden, sodass die genaue Bestimmung des Kontextes hinter diesen Werten möglich ist [24, 19].
Die zuvor beschriebenen Arten der Heterogenität werden in Tabelle 1 zusammengefasst. Diese Arten der Heterogenität stellen das Hauptproblem bezüglich der Zusammenführung von Datenquellen dar.

Heterogenität	**Bedeutung**
Technische	Technische Realisierung des Zugriffs auf Daten
Datenmodell	Datenmodell zur Repräsentation der Daten
Schematische	Spezialfall der strukturellen Heterogenität; verwendete Schemaelemente
Strukturelle	Strukturelle Repräsentation der Daten
Syntaktische	Darstellung von Informationen
Semantische	Bedeutung der Daten bzw. verwendeten Begriffe

Tabelle 1: Zusammenfassung der Arten von Heterogenität [24]

3.1.2. Arten von Integration

Bei der Integration werden zwei Arten unterschieden, zum einen die materialisierte und zum anderen die virtuelle Integration, die sich jeweils auf den Speicherort der Daten beziehen.

Materialisierte Integration

Bei der materialisierten Integration verbleiben die zu integrierenden Daten jeweils in den Datenquellen und werden zusätzlich in das integrierte System übertragen und dauerhaft, beispielsweise mittels eines Datenbankmanagementsystem (DBMS) gespeichert. Die Daten werden in die Zielstruktur transformiert. Die Daten sind im Integrationssystem *materialisiert* vorhanden. Diese Art der Integration findet sich vor allem bei Data-Warehouses wieder. Der Vorteil dieser Integrationsart liegt bei den Anfragen, die aus den integrierten Daten effektiv bedient werden können. Zusätzlich können aufwändige Transformationen zwischen dem globalen[18] und dem lokalen[19] Schema durchgeführt werden. Die Daten liegen zentral vor und die Anfragen werden direkt an das globale Schema gestellt. Vorteil hier sind die hohen Geschwindigkeiten der Anfragen, da nicht jede Datenquelle angesprochen werden muss, sondern nur das globale System.
Der Nachteil dieser Integrationsart sind die hohen Speicheranforderungen, die durch die zentrale Speicherung der Daten verursacht werden. Außerdem sind die integrierten Datenbestände nicht zu jeder Zeit auf dem aktuellsten Stand, sondern werden durch ein Update im Integrationssystem aktualisiert. Werden Updates im Quellsystem durchgeführt, dann sind diese aktualisierten Daten nicht gleichzeitig im Integrationssystem aktuell, sondern von einer Updatefrequenz abhängig [24, 31].

Virtuelle Integration

Bei der virtuellen Integration werden die Daten nicht zentral gespeichert, sondern verbleiben in den jeweiligen Datenquellen bzw. Quellsystemen. Die Daten sind demnach nicht materialisiert im Integrationssystem vorhanden, sondern *virtuell*.
Hierbei werden lediglich zum Zeitpunkt der Anfragebearbeitung die Daten teilweise aus den Quellsystemen in das Integrationssystem geführt und anschließend wieder zurückgewiesen. Die Integration erfolgt ausschließlich bei jeder Anfrage und werden bei jedem Zugriff zusammengeführt.
Der Vorteil dieser Integrationsart liegt bei der Aktualität der Datenbestände, die aufgrund der direkten Anfrage von den Datenquellen in das Integrationssystem übertragen werden. Werden Updates der Daten im Quellsystem durchgeführt, dann werden die aktualisierten Daten gleichzeitig in das Integrationssystem übertragen. Ein weiterer Vorteil ist bezüglich des Speicherbedarfs festzustellen. Da die Daten nicht im Integrationssystem zentral gespeichert werden, sondern lediglich geringer Speicherbedarf für die Ergebnisse der Anfragen benötigt wird.

[18] Das globale Schema ist die Vereinigung der Quellschemata zu einem einheitlichen Schema und wird häufig auch als Zielschema bezeichnet.
[19] Das lokale Schema ist das Schema der Quelldaten und wird auch als Quellschema bezeichnet.

Der Nachteil dieser Integrationsart ist die Performance. Da die Datenbestände jeweils bei der Anfrage transportiert werden, hängt die Antwortzeit unter anderem von den Zugriffsgeschwindigkeiten der Datenquellen und den Übertragungswegen ab. Diese Abhängigkeit wirkt sich negativ auf die Geschwindigkeit aus [24, 31].

3.1.3. Integrationsschritte nach Haas

Das Ziel der Integration besteht darin, heterogene Quelldaten weitestgehend vollständig, effizient und gemeinsam in eine systematisch gegliederte Einheit zu bringen. Diese Integration von Daten aus heterogenen Quellen ist in mehreren aufeinander folgenden Prozessschritte eingeteilt, die Laura Haas in [18] charakterisiert.

Verständnis In dem ersten Prozessschritt geht es darum, die Daten die in ein neues System integriert werden sollen, umfangreich zu analysieren. Dabei müssen die relevanten Daten wie Schlüssel, Datentypen, Beziehungen etc. erkannt werden. Während dieser Phase ist eine statische Auswertung der vorhandenen Datensätze möglich. Dabei können unter anderem inkonsistente Werte sowie besonders häufig vorkommende Werte festgestellt werden. Die zu integrierenden Daten werden erfasst [32, 18].

Vereinheitlichung Die zweite Phase baut auf die erste Phase auf. Hierbei wird ausgehend von der Analyse der zu integrierenden Daten eine Festlegung getroffen, wie die Daten in einem neuen System repräsentiert werden. Das bedeutet konkret, dass in diesem Schritt das Zielschema entworfen wird. Mittels Regeln wird eine Festlegung zur Darstellung der Daten im neuen System getroffen. Ausgehend von dem Quellschema werden Abbildungen auf das Zielschema identifiziert und Entscheidungen getroffen, wie mit bestimmten unvollständigen und inkonsistenten Daten umgegangen werden soll. Hierbei wird die Frage geklärt, wie Datenkonflikte gelöst und redundante Daten im Zielschema abgelegt werden können. Ebenfalls relevant ist das Erkennen und Zusammenführen von Duplikate[20] [32, 18].

Spezifikation In der dritten Phase werden ausgehend von dem Quellschema Abbildungen zu dem Zielschema festgelegt. Dies zeigt, an welcher Position die Quelldaten in dem neuen System abgelegt werden. Daraufhin werden sogenannte Transformationsanfragen[21], die in einer bestimmten Sprache wie Structured Query Language (SQL) oder auch XQuery[22] beschrieben sind, abgeleitet [32, 18].

Ausführung Die letzte Phase besteht darin, die Integration der Quelldaten auszuführen. Dabei unterscheidet man zwischen der **materialisierten**- und **virtualisierten** Integration. Wird die Integration materialisiert ausgeführt, dann werden die zu integrierenden Quelldaten direkt im Zielsystem gespeichert. Erfolgt die Integration virtualisiert, dann werden lediglich Anfragen in einer bestimmten Sprache an die Quelldaten gestellt. Die benötigten Daten werden bei dieser Art der Integration nicht direkt gespeichert, sondern die Ergebnisse aus der gestellten Anfrage werden im Zielsystem abgelegt [32, 18].

[20] Ein Duplikat stellt eine Kopie dar, in diesem Fall sind speziell gleiche Objekte gemeint.
[21] Eine Transformationsanfrage überführt die Daten des Quellschemas in die Struktur des Zielschemas.
[22] XQuery ist eine Abfragesprache für XML Datenbanken.

3.1.4. Techniken der Datenintegration

Leser und Naumann stellen in [24] konkrete Techniken der Datenintegration vor. Diese Techniken widmen sich hauptsächlich der Überwindung struktureller und semantischer Heterogenität, die der Schemaebene[23] zuzuordnen sind. Diese Techniken werden unter dem Begriff *Schemamanagement* zusammen gefasst und behandeln den Umgang mit heterogenen Schemata. Das Ziel hierbei ist das Problem der Heterogenität zu überwinden und gleichzeitig die Datenbestände in eine gemeinsame Struktur zu überführen.
Ein wichtiges Konzept stellen die *Korrespondenzen*, auch *Mappings* genannt, dar. Diese werden in [24, S. 115f] wie folgt definiert:

> *„Ein Mapping beschreibt einen semantischen Zusammenhang zwischen Elementen verschiedener Schemata. Mappings werden entweder per Hand erstellt, oder mittels Schema Matching semiautomatisch ermittelt. “*

Die nachfolgend betrachteten Techniken werden vor allem in der Phase *Spezifikation* sowie *Ausführung* der Integrationsschritte nach *Haas L.* in [18] eingesetzt [32]. Das aus den ersten beiden Phasen entstandene Zielschema sowie die definierten Korrespondenzen werden als Grundlage für diese Techniken angesehen.

Schemaintegration

Die Aufgabe der Schemaintegration ist das Zusammenführen von lokalen, heterogenen Schemata in ein einheitliches und globales Schema. Die Schemaintegration ist sowohl auf semantische, als auch auf strukturelle und syntaktische Unterschiede der lokalen Schemata zurückzuführen. Die Schemaintegration verfolgt vier Ziele, die wie folgt definiert werden:

1. **Vollständigkeit:** Das globale Schema muss sämtliche Konzepte der lokalen Schemata umsetzten.
2. **Minimalität:** Das globale Schema sollte gleiche Konzepte der lokalen Schemata vereinen und als ein Konzept darstellen.
3. **Korrektheit:** Das globale Schema darf keine Widersprüche zu den lokalen Schemata aufweisen, die Konzepte sollten sowohl im globalen, als auch in den jeweiligen lokalen Schemata die gleiche Bedeutung aufweisen.
4. **Verständlichkeit:** Das globale Schema muss sowohl für die Anwender, als auch für die Entwickler allgemein verständlich sein [24].

Da die Schemaintegration nicht zwingend automatisch erfolgen kann, existieren Vorgehensweisen mit deren Hilfe die Schemaintegration durchgeführt werden kann. Der erste Schritt ist die *Vorintegration*. In diesem Schritt wird die Reihenfolge für die Integration festgelegt sowie die lokalen Schemata selektiert. Dabei können alle vorhandenen lokalen Schemata zugleich oder jeweils ausgewählte lokale Schemata integriert werden. Der zweite Schritt beschreibt den *Schemavergleich*.

[23] Die Schemaebene betrachtet die Elemente des verwendeten Datenmodells für die Modellierung des jeweiligen Sachverhaltes.

In diesem Schritt werden Korrespondenzen zwischen semantisch gleichen Elementen der Schemata aufgestellt. In diesem Schritt werden ebenfalls strukturelle sowie semantische Probleme ermittelt. Diese Probleme werden im dritten Schritt, der *Schemaangleichung*, überwunden indem beispielsweise eine Umstrukturierung von Konzepten oder eine Umbenennung von Attributen der Schemata vorgenommen wird. Daraus resultieren jeweils transformierte Zielschemata. Diese transformierten Zielschemata werden im vierten Schritt, der *Schemafusion*, in ein einheitliches globales Schema fusioniert. In diesem Schritt können bezüglich des globales Schemas weitere Anpassungen vorgenommen werden [24, 8].
Es existieren verschiedene Integrationsverfahren, um die Schemaintegration strukturiert durchführen zu können. Im Folgenden wird eine weitere Technik des Schemamanagements erläutert. Diese Technik ist Bestandteil der vorliegenden Arbeit.

Schema Mapping

In diesem Abschnitt wird das Schema Mapping als Prozess beschrieben. Der Begriff beschreibt ebenfalls eine Menge von Korrespondenzen, die bereits definiert wurden und werden konkret als Wertkorrespondenzen bezeichnet. Der Schema Mapping Prozess leitet auf Basis der Wertkorrespondenzen unter anderem die Datentransformationsvorschriften ab.
Im Unterschied zur Schemaintegration, wo ebenfalls Korrespondenzen auftauchen und dabei für die Entwicklung eines neuen Schemas genutzt werden, werden beim Schema Mapping Prozess die zu integrierenden Daten zwischen bereits vorhandenen Schemata transformiert [24, 25]. Voraussetzung für das Schema Mapping sind sowohl das Quell-, als auch das Zielschema. Das Quellschema beschreibt ein lokales Schema einer Datenquelle und das Zielschema beschreibt ein globales Schema. Jedes Quellschema beinhaltet eine Datenmenge. Dies ist in Abbildung 10 illustriert. Die Datenmenge des Quellschemas bezeichnet die Quelldaten. Diese Quelldaten sollen in das Zielschema transformiert werden. Die integrierten Daten bilden anschließend die Zieldaten des Zielschemas [8].

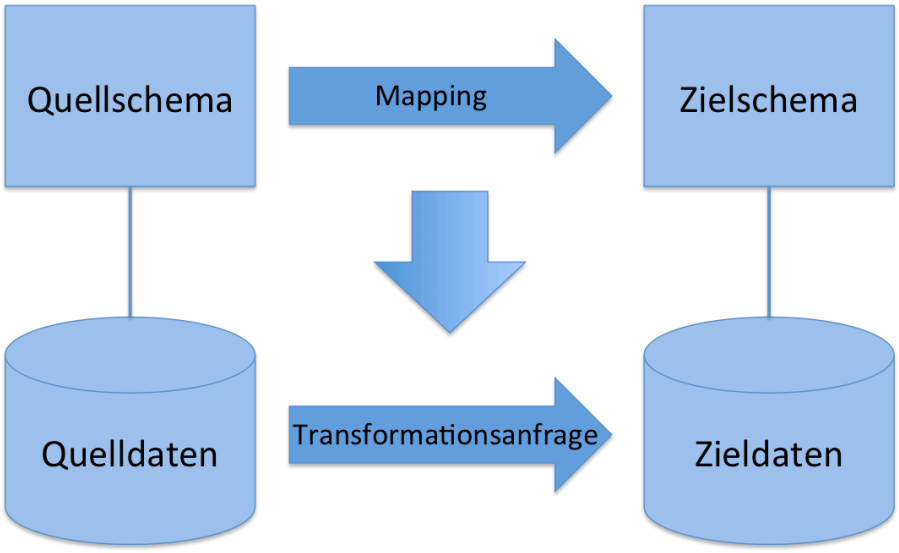


Abbildung 10: Verwendung von Schema Mappings; Datentransformation [25]

Der Schema Mapping Prozess ist in Abbildung 11 illustriert. Zu Beginn ist das Aufstellen von Wertkorrespondenzen zwischen den Schemaelementen notwendig. Über Wertkorrespondenzen wird festgelegt, wie die Werte der Attribute des Zielschemas aus den Werten der Attribute des Quellschemas generiert werden [24]. Der nächste Schritt besteht darin, die zuvor generierten Wertkorrespondenzen zu interpretieren. Hierbei werden die Wertkorrespondenzen in ein oder mehrere logische Mappings übersetzt. Ein logisches Mapping ist demzufolge das Resultat dieser Interpretation. Zuletzt müssen aus den logischen Mappings die Transformationsanfragen abgeleitet werden. Über diese Anfragen werden die Daten des Quellschemas in die Struktur des Zielschemas überführt bzw. zwei Schemata miteinander verknüpft. Anfragesprachen sind unter anderem SQL oder XQuery. Für die Transformationsanfrage wird häufig der Begriff *Schemakorrespondenz* verwendet [19, 24]. Der letzte Schritt ist die eigentliche Transformation der Daten. Die Quelldaten werden in die Struktur des Zielschemas transformiert. Hierbei kann, wie bereits in Abschnitt 3.1.2 beschrieben, zwischen materialisierte und virtuelle Integration unterschieden werden. Bei der materialisierten Integration kann die generierte Transformationsanfrage direkt ausgeführt werden. Bei der virtuellen Integration werden die Anfragen erst dann ausgeführt, wenn die Daten benötigt werden.

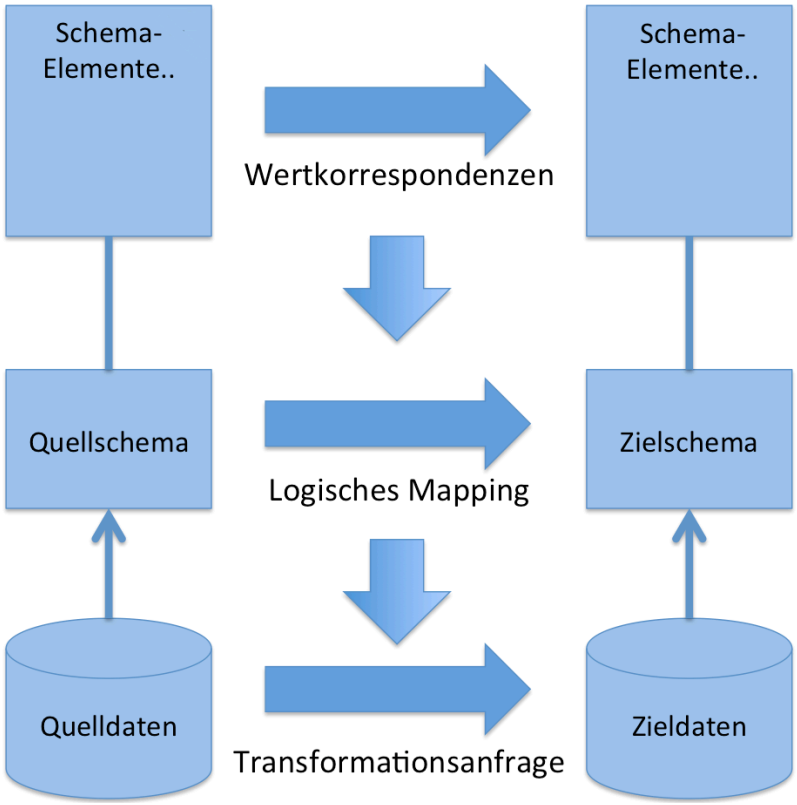


Abbildung 11: Schema Mapping Prozess [24]

Schema Matching

Die bereits beschriebenen Techniken, Schemaintegration und Schema Mapping, sind von einer Menge von Korrespondenzen abhängig. Diese Korrespondenzen werden zwischen Schemaelementen spezifiziert. Das Erstellen dieser Korrespondenzen kann manuell erfolgen. Das Finden solcher Korrespondenzen zwischen Schemaelementen ist ein aufwändiger Prozess, der ebenfalls Probleme mit sich führt. Die Auswirkungen solcher Probleme führen häufig zu fehlerhaften oder falschen Korrespondenzen.
Mit Hilfe von verschiedenen Verfahren können automatisch Korrespondenzen ermittelt und generiert werden. Diese Verfahren werden unter dem Begriff Schema Matching zusammen gefasst. Hierbei werden durch sogenannte *Matcher* die verschiedenen Schemata und unter anderem deren Struktur analysiert. Aus der Analyse werden Lösungen für Korrespondenzen vorgeschlagen, die manuell übernommen werden können [24].
Das Schema Matching sollte an dieser Stelle lediglich als weitere Technik des Schemamanagements erwähnt werde. Diese Technik nicht Bestandteil der vorliegenden Arbeit ist.

3.1.5. Anfragebearbeitung

Die Aufgabe der Anfragebearbeitung ist sämtliche Antworten auf die gegebene Anfrage an das globale Schema und der zusätzlich gegeben Menge von Korrespondenzen (Mappings) zwischen dem globalem und den lokalen Schemata zu finden [19]. Die Anfragebearbeitung spielt sowohl bei der materialisierten, als auch bei der virtuellen Integration eine entscheidende Rolle. Die Aufgabe der Anfragebearbeitung ist in fünf Schritten unterteilt, die im Folgenden definiert werden.

1. **Anfrageplanung:** Der erste Schritt der Anfragebearbeitung besteht darin, die Anfrage an das eigentliche globale Schema zu unterteilen bzw. zu zerlegen. In diesem Zusammenhang erfolgt die Übersetzung der globalen Anfrage in verschiedene lokale Anfragen. Hierzu wird die globale Anfrage unterteilt. Die aus der globalen Anfrage gebildeten Teilanfragen können jeweils von den einzelnen Datenquellen beantwortet werden. Anschließend müssen die einzelnen Datenquellen für die Beantwortung dieser Teilanfragen ausgewählt werden. Das heißt an dieser Stelle wird festgelegt, welche Teilanfragen an welche Datenquelle geschickt werden soll [19]. Hieraus ergeben sich Anfragepläne, die jeweils aus Teilanfragen bestehen. Das Ergebnis kann für die globale Anfrage zu einer Gesamtheit zusammengefasst werden [24, 19].

2. **Anfrageübersetzung:** In diesem Teilschritt werden die zuvor spezifizierten Teilanfragen in lokale Anfragesprachen übersetzt. Das bedeutet, dass aus der globalen Anfragesprache, lokale Anfragesprachen überführt werden, die jeweils von den Datenquellen verstanden werden können. Für jeden Anfrageplan ergibt sich somit eine Reihe von unterschiedlichen Teilanfragen, die tatsächlich ausgeführt werden können [24].

3. **Anfrageoptimierung:** Im dritten Schritt wird eine geeignete Reihenfolge der auszuführenden Teilanfragen festgelegt. In diesem Zusammenhang muss eine optimale Ausführungsreihenfolge bestimmt werden. Das Resultat ist ein Ausführungsplan [19].

4. **Anfrageausführung:** In diesem Schritt werden die spezifizierten Teilanfragen in der zuvor festgelegten Reihenfolge ausgeführt. Die übersetzten lokalen Anfragen werden an die jeweiligen Datenquellen gesendet und dort ausgeführt. Die Ergebnisse der Teilanfragen an die Datenquellen sind Teilergebnisse bzw. Teilantworten [24, 19].

5. **Ergebnisintegration:** Die aus dem vorherigen Schritt erhaltenen Teilantworten werden in diesem Schritt zu einem einheitlichen Gesamtergebnis im Integrationssystem zusammengeführt [24, 19].

Die Schritte zwei bis fünf können unter anderem durch ein Programm automatisiert werden. Für den ersten Schritt, der Anfrageplanung, wird jedoch semantisches Wissen benötigt. In diesem Zusammenhang muss festgelegt werden, durch welche Datenquellen die unterschiedlichen Einheiten der globalen Anfrage beantwortet werden können [24]. Hierbei können sogenannte Anfragekorrespondenzen genutzt werden, mit deren Hilfe semantische Beziehungen zwischen dem globalen und den lokalen Schemata festgestellt werden können.

3.1.6. Integrationsarchitekturen

Für die technische Realisierung der Integration von verteilten heterogenen und autonomen Informationssystem sind unter anderem Techniken notwendig, mit deren Hilfe die Integration der Daten realisiert werden kann. Für die Integration auf Datenebene[24] existieren unterschiedliche Architekturen, die im Folgenden erläutert werden.

Extraktion-Transformation-Laden

Die bekannteste Architektur der Integration auf Datenebene stellt der Extraktion-Transformation-Laden (ETL) Ansatz dar. Hierbei werden die Datensätze aus vorhandenen Systemen vorab extrahiert und gemeinsam in ein neues System geladen. Während dieses Prozesses werden die Daten, die aus heterogenen Datenquellen stammen, weitestgehend homogenisiert und über entsprechende Transformationsvorschriften realisiert. Das Ergebnis ist eine materialisierte Speicherung der Datenbestände in dem Zielsystem.
Der erste Schritt des ETL Prozesses behandelt die *Extraktion*. Die relevanten Daten, die in operativen Datenbanken, Flatfiles oder anderen Dokumenten gespeichert sind, werden extrahiert[25]. Die Daten können strukturiert, semistrukturiert oder unstrukturiert vorliegen. Strukturierte Daten liegen meist in Datenbanken oder Tabellen vor, unstrukturierte in E-Mails oder Briefen und semistrukturierte z.B. in XML- oder CSV-Dateien. Für die Extraktion dieser Daten werden verschiedene Techniken angewendet und die jeweiligen Vorteile der Speicherung genutzt. Dabei ist es wichtig, dass ein Zugriff auf die vorliegenden Daten erfolgen kann.

[24] Die Integration auf Datenebene beschreibt die Datenerhaltungsschicht, die der permanente Speicherung der relevanten Daten dient.
[25] Bei der Extraktion werden die relevanten Daten herausgelesen.

Bei der *Transformation* werden die zuvor extrahierten Daten schrittweise vereinheitlicht. Dieser Schritt ist für die Homogenisierung der Daten zuständig und als wichtiger Schritt des ETL Prozesses anzusehen. Die *syntaktische* Transformation behandelt die Probleme unter anderem hinsichtlich Datentypen, Datenformate sowie Zeichensatz. Beispielsweise wird durch eine Vereinheitlichung des Datenformates versucht, die Probleme zu lösen. Bei der *strukturellen* Transformation werden die Datenbestände auf Modellebene betrachtet. Dabei wird unter anderem durch die Einführung von Schlüsselwerten eine Vereinheitlichung erreicht.
Bei der *semantische* Transformation wird durch eine einheitliche Namensgebung, z.B. beim Längenmaß, versucht, eine Vereinheitlichung zu erreichen.
Nach Abschluss der ersten beiden Schritte sind alle relevanten Daten für den dritten Schritt, das *Laden*, bereit. Das Ziel hierbei ist die Daten permanent im Zielsystem abzulegen und für weitere Anwendungen zur Verfügung zu stellen. Das Laden der transformierten Daten in das Zielsystem stellt einen langwierigen Prozess dar, welcher von verschiedenen Faktoren abhängig ist. Zum einen vom Umfang und zum anderen von der Komplexität der zu integrierenden Daten [28].

Extraktion-Laden-Transformation

Bei dem Extraktion-Laden-Transformation (ELT) Prozess werden die Teilschritte in einer anderen Reihenfolge als bei ETL ausgeführt. Der erste Schritt besteht ebenfalls darin, die erforderlichen Daten zu extrahieren. Anschließend werden in diesem Prozess allerdings die Daten sofort in das Zielsystem geladen bzw. kopiert. Die Transformation wird nicht wie bei ETL als zweites ausgeführt, sondern erst auf spezielle Anfrage direkt auf dem Zielsystem ausgeführt. ELT beschreibt in der Datenintegration den modernen, währenddessen ETL den klassischen Ansatz verfolgt [28].

3.2. Datenbankkonzeption

In diesem Abschnitt werden die Grundlagen der Datenbankmodellierung näher erläutert. Hierzu werden die Phasen des Datenbankentwurfs sowie anschließend das Entity-Relationship-Modell (ER-Modell) und das relationale Datenbankmodell vorgestellt.
Um bezüglich der Datenintegration ein geeignetes Zielschema zu entwerfen, sind Kenntnisse in der Modellierung von Datenbanken notwendig. Mit Hilfe eines Datenbanksystems können Daten elektronisch gespeichert und verwaltet werden. Ein Datenbanksystem beinhaltet zum einen eine Datenbank und zum anderen ein DBMS. Die gesamten Softwaremodule, welche für die Verwaltung der Datenbank zuständig sind, werden unter dem Begriff DBMS zusammengefasst [29, S. 8]. Eine Datenbank bildet die Basis für die Speicherung der Daten, die jeweils vom Datenbanksystem verwaltet werden.

3.2.1. Einführung in den Datenbankentwurf

Um eine Datenbankanwendung konzipieren und umsetzen zu können, müssen die Schritte des Datenbankentwurfs durchlaufen werden. In Abbildung 12 sind die vier grundlegenden Phasen des Datenbankentwurfs abgebildet.

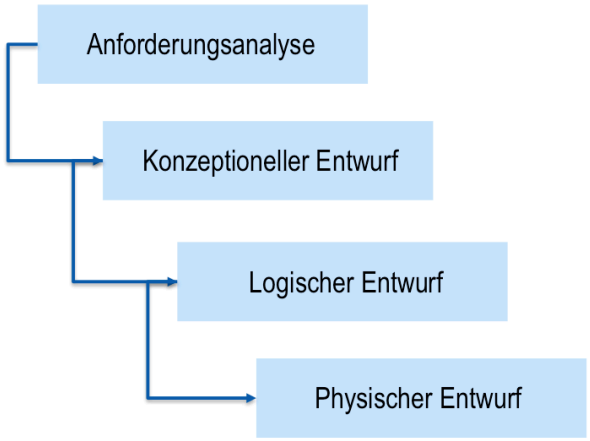


Abbildung 12: Die vier Entwurfsphasen

In der ersten Phase, der Anforderungsanalyse, werden Anforderungen und der eigentliche Sinn und Zweck der Datenbank analysiert. Dabei wird spezifiziert, welche Aspekte abgebildet werden müssen. Das Ergebniss dieser Phase ist im allgemeinen eine informelle Beschreibung der vorliegenden Anwendung bzw. des Datenbedarfs [29].
In dem konzeptionellen Entwurf, der zweiten Phase der Datenbankmodellierung, werden die in der erste Phase gewonnenen Erkenntnisse unabhängig von der konkreten Implementierung in einem Datenbankentwurf dargestellt. Hierzu erfolgt eine formale Beschreibung der Anwendung. Dabei existieren zahlreiche Modellierungsverfahren, wie z.B. das Entity-Relationship (ER)-Modell.

In der dritten Phase, dem logischen Entwurf, wird ein konkretes Datenbankmodell, z.B ein relationales- , ein objektorientiertes- oder ein hierarchisches Datenbankmodell, gewählt. In dieser Phase wird das zuvor entworfene Modell in ein konkretes Datenbankmodell überführt. Mit Hilfe eines Datenbankmodells werden die Strukturen für die jeweils zu verarbeitenden Daten definiert. In dieser Phase kann ebenfalls ein konkretes Datenbanksystem, z.B. MySQL[26] gewählt werden.
Der physische Entwurf stellt die letzte Phase in der Datenbankmodellierung dar. In dieser Phase erfolgt die Beschreibung der Speicherstrukturen. In diesem Zusammenhang wird die Art und Weise der Datenspeicherung festgelegt [29, 26].
Die Datenbankmodellierung stellt eine große Herausforderung dar. Das Ziel der Modellierung ist ein wartbares und erweiterbares Datenbankmodell, das durch einen Datenbankentwickler festgelegt wird.

3.2.2. Das ER-Modell

Eines der bekanntesten Entwurfsmodelle ist das ER-Modell, über welches Ausschnitte der realen Welt beschrieben werden können. Das ER-Modell wurde erstmals von Peter Chen 1976 vorgestellt.

Mit der Zeit haben sich eine Reihe an Modellierungsvarianten und Erweiterungen für das ER-Modell entwickelt. Ein einheitlicher Standard existiert bisher nicht. Das ER-Modell ist nicht von einem bestimmten Datenbanksystem abhängig, sondern kann für unterschiedliche Datenbanksysteme verwendet werden [29, 21].
Das ER-Modell setzt sich aus den Modellierungsstrukturen *Entities*, *Beziehungen* und *Attribute* zusammen. Diese Strukturen werden im Folgenden definiert.

1. **Entity**: Eine Entity ist ein Objekt der realen Welt, über welches Informationen gespeichert werden sollen. Eine Entity ist die zu modellierende Informationseinheit [29].
2. **Beziehungen**: Beziehungen werden zwischen den Entities modelliert. Dabei existieren unterschiedliche Arten von Beziehungen.
3. **Attribute**: Attributen sind Eigenschaften von Entities und Beziehungen.

Die Entities werden anhand von Attributen beschrieben, die Eigenschaften des realen Objektes darstellen und müssen eindeutig identifiziert werden. Die Entities werden in Beziehung zueinander gestellt, über die jeweils Abhängigkeiten und Zusammengehörigkeiten zwischen den Entities ausgedrückt werden. Beziehungen können anhand von Attributen näher beschrieben werden. Mittels der Angabe von Kardinalitäten wird die Menge der Entities festgelegt, die jeweils an einer Beziehung teilnehmen.

[26] MySQL ist eines der am weitesten verbreiteten relationalen DBMS

Es existieren verschiedene Arten von Beziehungen.

- **1:1**; Jede Entity eines Entity-Typ ist genau einer Entity eines anderen Entity-Typs zugeordnet [29].
- **1:n**; Jede Entity eines Entity-Typ werden beliebig viele Entities eines anderen Entity-Typs zugeordnet [29].
- **n:m**; Mehrere Entities eines Entity-Typ können beliebig viele Entities eines anderen Entity-Typs zugeordnet werden. [29].

Weiterhin existieren eine Reihe verschiedener Modellierungsformen. In diesem Zusammenhang hat sich auch die Chen-Notation entwickelt, die sich auf die zuvor aufgelisteten Arten von Beziehungen beziehen. Eine weitere Form der Notation ist die Min/Max-Angabe. Diese Notationsform ist Bestandteil der vorliegenden Arbeit und wird in Kapitel 5 für die Modellierung der Datenbank eingesetzt und detailliert beschrieben.
In der vorliegenden Arbeit wird das ER-Modell als Grundlage für die Implementierung der zu modellierenden Datenbank genutzt und wird in Kapitel 5.1.1 anhand eines konkreten Beispiels erläutert.

3.2.3. Das relationale Datenbankmodell

Das relationale Datenbankmodell ist eines der weltweit am weitesten verbreiteten Datenbankmodelle und wurde erstmals von E.F. Codd im Jahre 1970 formuliert. Das relationale Datenbankmodell, das Teil der logischen Datenbankdesign-Schicht ist, bildet unter anderem die Grundlage von vielen DBMS.
Hauptbestandteil des relationalen Datenbankmodells sind die Faktoren Relation, Attribut und Tupel.

Relation: Eine Relation bezeichnet eine Tabelle in einer Datenbank.

Attribut: Ein Attribut beschreibt eine Spalte in einer Tabelle.

Tupel: Ein Tupel ist eine Zeile, konkret ein Datensatz, einer Tabelle.

Die zentralen Elemente bilden die Relationen, die jeweils eine Anzahl von Attributen beinhalten. Die Anzahl der Tupel kann hingegen variieren. Über eine Relation werden Daten zueinander in Beziehung gesetzt. Eine Relation ist eine zweidimensionale Datenstruktur [26]. Diese Datenstruktur besteht aus Zeilen und Spalten. Eine Tabelle ist durch eine eindeutige Bezeichnung, auch Tabellenname genannt, gekennzeichnet. Notwendig ist die Eindeutigkeit des Tabellennamens innerhalb einer Datenbank. Die Attribute einer Tabelle müssen ebenfalls einen eindeutigen Namen aufweisen, damit eine Spalte innerhalb einer Tabelle eindeutig identifiziert werden kann. Jedes Attribut steht für eine Eigenschaft eines Objektes, welches durch die Relation abgebildet wird. Ein Tupel bildet ein konkretes Objekt aus der realen Welt innerhalb der Datenbank [26, 21]. Ein Tupel muss innerhalb einer Datenbank eindeutig identifiziert werden können und muss somit von anderen Tupel voneinander unterscheidbar

sein. Durch die Eindeutigkeit der Datensätze kann gewährleistet werden, dass unter anderem keine Inkonsistenzen, also keine Widersprüche sowie Verwechslungen innerhalb der Datenbank, auftreten. Um einen eindeutigen Datensatz innerhalb einer Datenbank zu erzeugen, werden *Schlüssel* verwendet. Ein Schlüssel identifiziert ein Tupel eindeutig durch einen Wert oder durch eine Kombination von mehreren Werten. Die Kombination der Werte der Schlüsselattribute dürfen dabei nur einmal vorkommen. Hierbei werden ein oder mehrere Attribute einer Tabelle so gewählt werden, dass diese als eindeutige Identifikation eines Datensatzes genügen. Diese Schlüsselattribute werden als *identifizierende Attributmenge* bezeichnet. Ein Beispiel für einen künstlich eingeführten Schlüssel sind Zahlen, die automatisch beim Hinzufügen eines Datensatzes hochgezählt und vergeben werden. Hierbei wird jedem Datensatz eine eindeutige Nummer zugewiesen [26, 21].
In einem relationalen Datenbankmodell wird zwischen Primär- und Fremdschlüssel unterschieden. Eine Relation beinhaltet genau einen Primärschlüssel und ist für die eindeutige Identifizierung eines Tupels zuständig. Ein Fremdschlüssel wird für die Verknüpfung von unterschiedlichen Relationen miteinander verwendet. Ein Fremdschlüssel in einer Relation, welcher ein Attribut oder eine Attributmenge darstellen kann, ist in einer anderen Relation als Primärschlüssel gekennzeichnet. Weiterhin können jeweils zwischen zwei Relationen Bedingungen der *referentiellen Integrität* definiert werden. Über diese Bedingungen wird die Integrität zwischen den Tupeln, bezüglich der jeweiligen Relationen gewährleistet. Die Integritätsbedingungen werden unter anderem über die Festlegung von Schlüsseln formuliert. Die Einhaltung dieser Bedingungen ist zwingend notwendig [26].

Datenbankmanagementsystem MySQL

Das DBMS MySQL setzt das relationale Datenbankmodell ein und findet in der vorliegenden Arbeit als konkretes System Einsatz. MySQL ist ein weit verbreitetes relationales DBMS von *Oracle* erhältlich [26]. Auf dem MySQL-Server werden die Daten gespeichert und über die MySQL-Clients werden Anfragen an den Server gesendet. Der MySQL-Server ist das eigentliche DBMS, auf dem eine Reihe an Datenbanken implementiert werden können. Das hierfür grafische Entwicklungssystem *MySQL Workbench* ist ein visuelles Werkzeug. Die Workbench bietet unter anderem die konkrete Modellierung einer Datenbank an. In diesem Zusammenhang können visuell unterschiedliche Arten von Datenbanken entwickelt, modelliert und verwaltet werden sowie komplexe ER-Diagramme erstellt werden. Die MySQL Workbench wird in der vorliegenden Arbeit für die Modellierung der Datenbank verwendet.

Datenbanksprache SQL

Die Datenbanksprache SQL wird in relationalen Datenbanken hauptsächlich zur Abfrage von Datenmengen eingesetzt. Mit Hilfe von SQL können zusätzlich Datenstrukturen definiert und Datenbestände bearbeitet werden. SQL kann unabhängig vom Betriebssystem verwendet werden. Für die Datenintegration können mittels der Datenbanksprache SQL, Anfragen an die Datenbank spezifiziert werden.

3.3. Analyse von Excel-Tabellen

In diesem Anschnitt werden zwei Ansätze vorgestellt, mit deren Hilfe die Analyse von Excel-Tabellen ermöglicht wird.
Die Eingabe und Verarbeitung von komplexen numerischen sowie alphanumerischen Datensätzen lässt sich problemlos über eine Tabellenkalkulation, wie Microsoft Excel, realisieren. Ebenfalls erlaubt diese Art von Anwendungsprogrammen weitreichende Funktionen, wie die grafische Darstellung, Berechnungen über Formeln etc.
Tabellen stellen einen wichtigen Bestandteil in der Repräsentation von Daten dar. Informationen werden veranschaulicht dargestellt und darüber hinaus Zusammenhänge untereinander auf einer speziellen, kompakten Art und Weise dargestellt. Daten und Inhalte können zueinander in Beziehung gesetzt werden, die dabei eine strukturierte Sicht aufweisen.
Eine Tabelle ist als Ansammlung von Zellen anzusehen, die in einem 2D Gitter angeordnet sind. Jede dieser Zellen kann eindeutig über eine Adresse identifiziert werden und besitzt zudem einen Zeilen-, Spaltenindex und einen Wert. Die Zelle ist der Schnittpunkt zwischen einer Zeile und einer Spalte. Der enthaltene Wert ist entweder eine Konstante oder ein über eine Formel berechneter Wert. Eine Tabellenkalkulation kann eine Reihe von Tabellenblätter enthalten.
Mit der Analyse und Interpretation von Tabellen beschäftigt sich die Forschungsarbeiten von Doush I. und Pontelli E. in [11]. Ein Modell für die Definition von Tabellenstrukturen wird von Wang X. in [36] vorgestellt. Es existiert derzeit kein expliziter Mechanismus um alle Probleme einer Tabelle zu erfassen und zu verstehen. Doch werden in dieser Arbeit Bestrebungen beschrieben, um solche Tabellen analysieren zu können.

3.3.1. Ansätze von Doush und Pontelli

Eine Tabelle ist eine Ansammlung von funktionalen Komponenten, die in einer Baumstruktur dargestellt werden. Das Erfassen dieser Tabellen kann unter anderem mit Hilfe verschiedener Merkmale wie Hintergrundfarbe, Schriftart, Rahmen, leere Zeilen oder Spalten untersucht werden. Diese Merkmale werden genutzt um eindeutig zu identifizieren, welche Ansammlungen von Zellen eine Tabelle aufweisen.
Um den Inhalt einer Tabelle identifizieren zu können, ist eine konkrete Analyse notwendig. Diese Analyse gliedert sich in zwei Phasen:

1. Identifizieren einer Tabelle
2. Verstehen der funktionalen Komponenten einer Tabelle

Während dieser zwei Phasen werden sowohl Layout, als auch Sprache der Tabelle betrachtet. Für die Analyse der Struktur des Layouts werden Trennsymbole wie Leerzeichen, horizontale oder vertikale Linien, komplette leere Spalten oder Zeilen verwendet. Mit deren Hilfe können zusätzlich voneinander abgrenzende Tabellen bestimmt werden.

Der von [11] definierte Algorithmus nutzt sowohl gleiche Formatierungen, Trennsymbole und Ansätze von Wang um unterschiedliche funktionale Komponenten einer Tabelle zu bestimmen. Dieser Algorithmus wird im Folgenden näher erläutert.

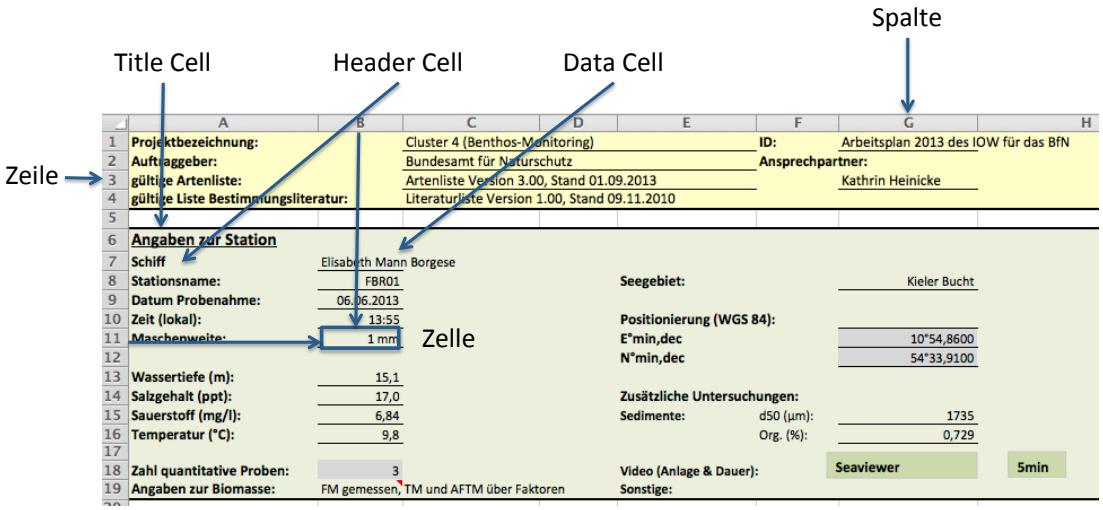


Abbildung 13: Komponenten einer Tabelle [11]

Die Eigenschaften einer Zelle in einer Tabellenkalkulation und deren Beziehung zu den umliegenden Zellen können genutzt werden, um funktionale Komponenten einer Tabelle zu erfassen. Der Umfang einer Tabelle ist abhängig von den vorhandenen Informationen. In der Abbildung 13 sind die relevanten Elemente einer Tabelle in einer Tabellenkalkulation aufgezeigt. Diese Elemente werden von Doush und Pontelli [11] verwendet, um eine Tabelle erfassen und verstehen zu können.

Eine Tabelle beinhaltet drei unterschiedliche Zelltypen, die im Folgenden klassifiziert werden. Eine **Header Cell** stellt eine oder eine Menge von Zellen oberhalb einer Spalte dar. Diese Klasse der Zelle wird verwendet, um die Kategorie bzw. den Namen einer Spalte zu beschreiben. Dabei kann es vorkommen, dass diese Zellen zu einer Gruppe zusammengefasst werden kann. Die Informationen können für den Anwender genutzt werden, um den Inhalt einer Spalte zu verstehen. Diese Klasse der Zellen beinhalten vorwiegend einen String als Datentyp. Die visuelle Formatierung wie Rahmen, Schriftfarbe oder Schriftstil wird genutzt, um einen gleich bedeutenden Bereich von Zellen einzubetten. Dieser Bereich kann dadurch von anderen Bereichen abgegrenzt werden. Unterhalb einer Header Cell befindet sich in den meisten Fällen eine leere Zelle. Diese Zelle dient als Abgrenzung zu den dazugehörigen Data Cells, die im nächsten Abschnitt definiert werden. Diese leere Zelle ist nicht zu verwechseln mit einer allgemeinen Abgrenzung zu einer weiteren Tabelle. Hierfür ist eine Menge von leeren Zellen von Bedeutung.
Eine **Data Cell** beinhaltet den eigentlichen Wert, welcher der Header Cell zugeordnet ist. Diese Klasse kann einen Wert oder einen über eine Formel berechneten Wert enthalten. Häufig ist eine Menge von Data Cells in einer Tabelle zu finden, die genau zu einer Header Cell zugeordnet ist. Für die Identifikation dieser Gesamtheit sind sowohl die Start- und Endzeile, als auch die Start- und Endspalte von Bedeutung. Mit deren Hilfe kann der Umfang einer Spalte erfasst werden. Die Ansammlung von Data Cells wird als Datentabelle zusammengefasst. Diese Datentabelle stellt den Körper einer Tabelle dar.
Eine **Title Cell** bildet einen bestimmten Block und ist immer oberhalb einer Tabelle zu finden, meist in der ersten Zeile und Spalte. Dieser Zell-Typ beschreibt den Inhalt der gesamten Tabelle. Neben der Title Cell sind leere Zellen zu finden.

Algorithmus

Viele Eigenschaften wie die Struktur des Layouts, die Formatierung einer Zelle sowie die Werte innerhalb einer Zelle werden verwendet, um eine Tabelle identifizieren zu können. Trennzeichen sind ausschlaggebend, um zwischen Tabellen zu differenzieren. Die logische und physikalische Struktur wird mittels Muster und dem hierarchischem Clustering erfasst. Diese Punkte werden vereint und in einem Algorithmus zusammen gefasst.
Der erste Schritt in dem Algorithmus aus [11] besteht darin, die in der Tabellenkalkulation vorliegende erste Zelle zu überprüfen. Die erste Zelle bildet den Schnittpunkt der ersten Zeile und Spalte. Der Algorithmus ist an jener Stelle beendet, an der sich eine nichtleere Zelle befindet. Während des Durchlaufens der Zellen werden folgende Eigenschaften überprüft:

- **Formatierungen**
  Es werden unter anderem Rahmen und Farbe für jede Zelle identifiziert. Mit diesen Eigenschaften ist es möglich Überschriften, Spalten- und Zeilenanzahl festzustellen.

- **Trennzeichen**
  Über leere Zeilen und Spalten können semantisch unterschiedliche Bereiche voneinander abgegrenzt und bestimmt werden. Auch durch die unterschiedlichen Klassen von Zellen wie Data-, Header- und Title Cell sowie durch unterschiedliche Arten von Um-

randungen bzw. Rahmen oder auch Hintergrundfarbe einer Zelle bzw. eines gesamten Zellbereiches, können zur Identifikation von zusammengehörigen Bereichen beitragen.

Doush und Pontelli geben eine konkrete Notation für eine Zelle an [11].

$$\mathbf{C[i,j]} \tag{1}$$

i ist der Zeilen- und j der Spaltenindex, siehe dazu Abbildung 14. Jede Zelle kann einen Wert enthalten, welcher sowohl eine Konstante als auch ein über eine Formel berechneten Wert annehmen kann.

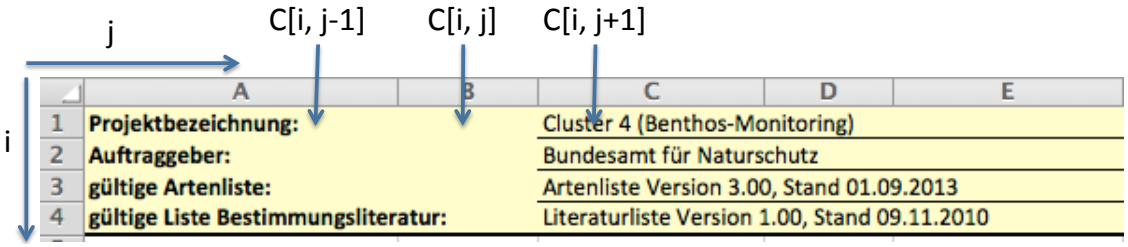


Abbildung 14: Positionierung einer Zelle anhand des jeweiligen Spalten- und Zeilenindex

Im Folgenden werden die für den Algorithmus notwendigen Merkmale einer Zelle spezifiziert. Diese umfassen *a) Rahmen*, *b) Formel*, *c) Format* und *d) Datentyp*. Jedes dieser Eigenschaften hat einen Wert, der über eine Notation ermittelt werden kann.

$$\mathbf{C[i,j].attribute\_name} \tag{2}$$

Für den Ausdruck *attribute_name* können Eigenschaften angegeben werden. Dadurch kann z.B. über *C[5,6].border.right.color* die Farbe des Rahmens links der Zelle in Zeile (6) und Spalte (5) ermittelt werden.

Eigenschaften einer Zelle

Eine Zelle hat bestimmte Eigenschaften, die im weiteren Verlauf dieses Abschnittes spezifiziert werden.
Der **Datentyp** einer Zelle wird mit Hilfe der folgenden Notation identifiziert. Der Wert in einer Zelle kann den Typ *double, string, boolean, empty, error* annehmen.

$$\mathbf{C[i,j].type} \tag{3}$$

Der **Rahmen** stellt eine umrandete Linie einer Zelle dar und beinhaltet Merkmale wie *Richtung, Stil, Farbe* und kann die Richtung *oben, unten, links, rechts* annehmen.

Das **Format** bezeichnet die visuelle Formatierung des Wertes innerhalb einer Zelle und beinhaltet unter anderem Merkmale wie *Schriftgröße, Schriftart, Schriftname, fett, kursiv, unterstrichen.*

Eine **Formel** ist eine mathematische Gleichung, die den Wert einer Zelle berechnen kann. Berechnungen können abhängig von anderen Zellen in einer Tabelle durchgeführt werden. Mit folgender Notation kann festgestellt werden, ob der beinhaltete Wert über eine Formel berechnet wurde.

$$\mathbf{C[i,j].formula.has} \tag{4}$$

Die eigentliche Formel lässt sich durch einen Zusatzausdruck angeben.

$$\mathbf{C[i,j].formula.text} \tag{5}$$

Um eine Analyse durchführen zu können, muss jede Zelle durchlaufen und nach den zuvor genannten Regeln überprüft werden.

Ablauf

Für den Algorithmus sind sowohl die genannten Eigenschaften mit deren Notation, als auch die bereits spezifizierte Klassifikation der Zellen ausschlaggebend. Sie werden verwendet, um Zellen zu funktionalen Komponenten zusammen zu fassen. In Abbildung 15 ist das Zusammenfassen der Zellen zu funktionalen Komponenten grafisch dargestellt. In einem Tabellenblatt können mehrere Tabellen enthalten sein, die über die Anwendung des Algorithmus aus [11] bestimmt werden können. Die erfassten Tabellen werden in die zuvor genannten Klassen einer Zelle eingeteilt.

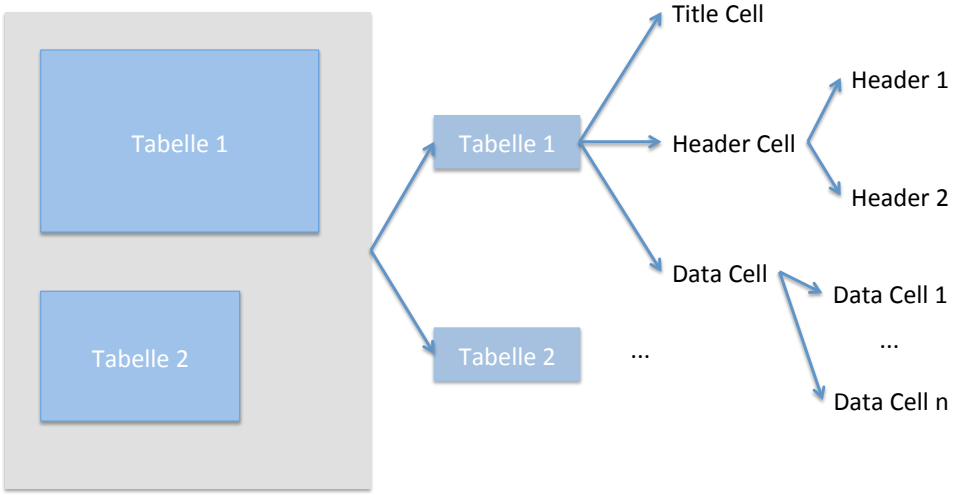


Abbildung 15: Zellen zu funktionalen Komponenten [11]

Der Algorithmus ist in vier Schritten abzuarbeiten. Diese Schritte werden genutzt, um Tabellen voneinander differenzieren zu können und z.B. in einem Tabellenblatt einer Tabellenkalkulation mehrere enthaltene Tabellen zu identifizieren.

1. **Start**
   Nichtleere gefundene Zellen werden zu T[27] hinzugefügt.

2. **Cluster**[28]
   Zellen werden gruppiert.

3. **Hinzufügen zu T**
   Zellen ebenfalls zu T hinzufügen.

4. **Stopp**
   Der Durchlauf ist an jener Stelle beendet, an welcher ein leerer Satz an Zeilen bzw. Spalten zu finden ist. Diese Zeilen und Spalten müssen innerhalb eines zuvor definierten Grenzbereiches liegen.

Nachdem die Schritte durchlaufen sind, ist es weiterhin notwendig, die unterschiedlichen Komponenten in der enthaltenen Tabelle zu erfassen und zu verstehen. Dieser Prozessschritt wird in der Abbildung 16 verdeutlicht. Das Flussdiagramm dient dazu, die Zellen hinsichtlich der Klassifikation zusammenzufassen. Es wird immer die aktuelle Zelle mit den zuvor spezifizierten Eigenschaften betrachtet. Anhand dieser Merkmale und Regeln ist zu bestimmen, ob die aktuelle Zelle einer Data Cell entspricht. Doush und Pontelli definierten Regeln mit deren Hilfe es möglich ist, die Zellen zu identifizieren und zu bestimmen, wann eine Zelle eine Header-, Title- oder Data Cell ist. Die Regeln werden in dieser Arbeit nicht verwendet und somit nicht in ihrer Form beschrieben. Ist nach der Überprüfung festzustellen, dass die aktuelle Zelle einer Data Cell entspricht, so ist diese zu der Menge von Data Cell hinzuzufügen. Ist dies nicht der Fall, dann wird das Flussdiagramm weiter durchlaufen und überprüft, ob eine Header- oder Title Cell vorliegt. Die erlangten Informationen der durchlaufenen Zellen werden für die weitere Nutzung der Tabelle gespeichert. Dadurch kann festgestellt werden, an welcher Position die Informationen zu einer gefundenen Header Cell liegen. Die Start- und Endzeile sowie die Anzahl der enthaltenen Data Cells einer Spalte kann somit genau bestimmt werden. Mit deren Hilfe kann gewährleistet werden, dass während der Analyse einer Tabelle die vorliegenden Werte richtig zugeordnet werden. Die Verwendung der drei unterschiedlichen Klassen einer Zelle entsprechend ihrer Funktion in einer Tabelle ist für den Algorithmus notwendig [11].

[27] T ist ein Satz von Zellen aus einer Tabelle
[28] Cluster bedeutet Gruppieren.

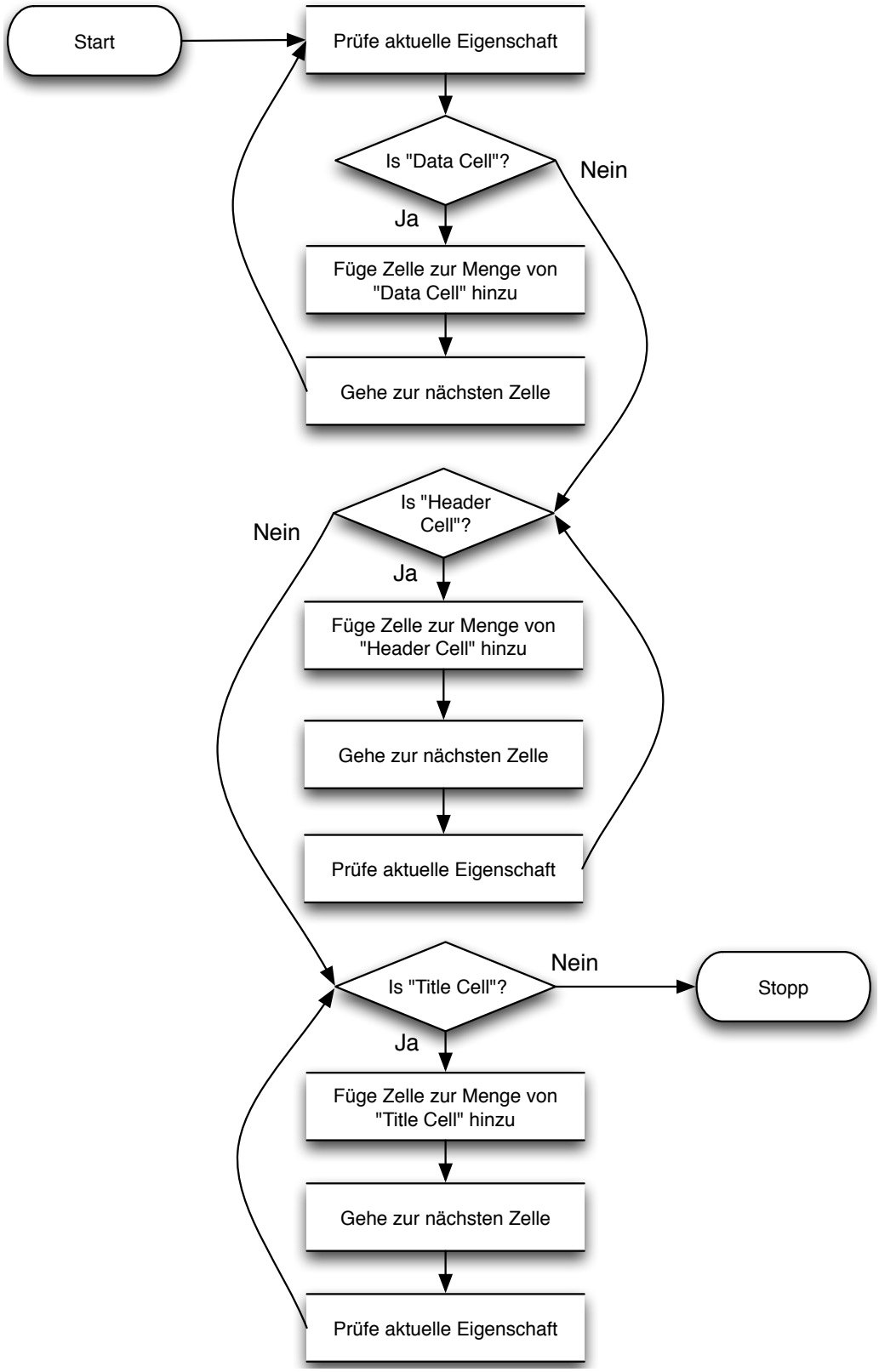


Abbildung 16: Identifikation der Zellen nach ihrer Klassifikation [11]

3.3.2. Definition der Tabellenstruktur von Wang

Die Hauptaufgabe einer Tabelle liegt darin, detaillierte Informationen in einer kompakten Art zu repräsentieren. Dadurch ist ein schneller Vergleich von Informationen möglich.
Eines der am weitesten verbreiteten Modelle für die Definition einer Tabellenstruktur wurde von Wang eingeführt. Wang identifiziert in [36] drei Abschnitte für die Struktur einer Tabelle. Zum einen den Zusammenhang zwischen Elementen, zum anderen die räumliche Darstellung des Inhaltes in einer Tabelle. Darunter zählen unter anderem Überschriften sowie Abschnitte. Der dritte Aspekt beschreibt die grafische Repräsentation des Inhaltes unter Verwendung von Schriftart und Zeichensatz sowie Trennzeichen und andere formatierende Merkmale.
Die von Wang in [36] definierten Bereiche, *Inhalt* sowie *Form* einer Tabelle, werden in dieser Arbeit verwendet und anhand der Abbildung 17 im weiteren Abschnitt beschrieben.

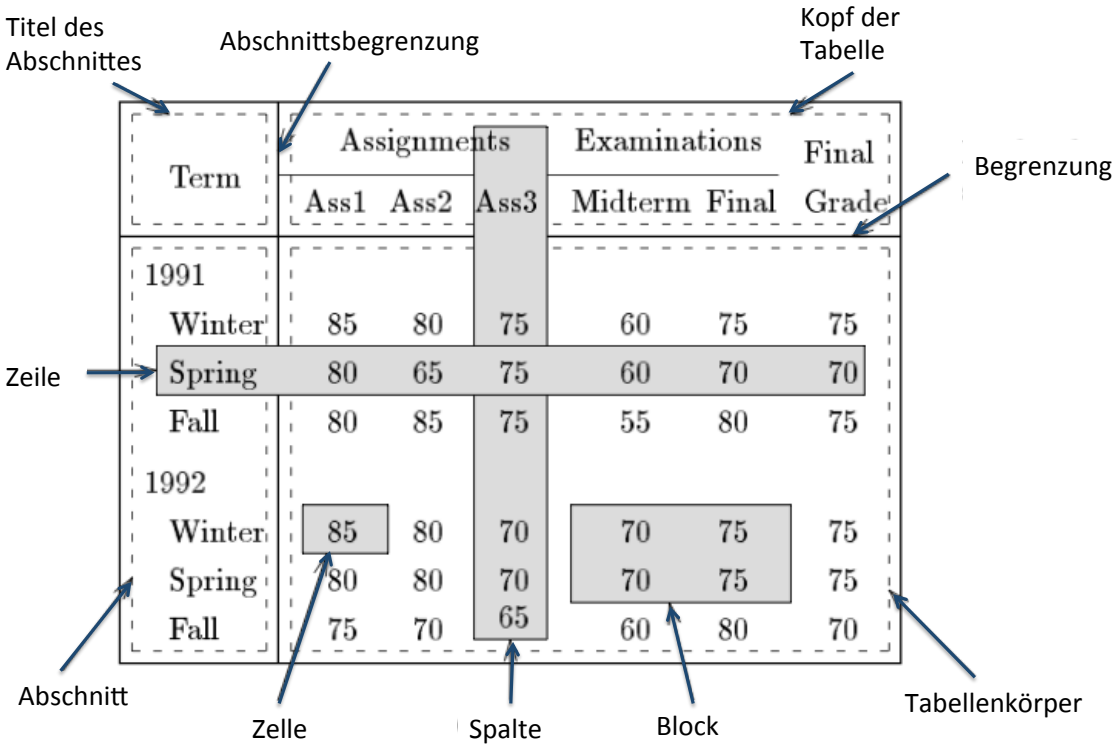


Abbildung 17: Strukturelle Komponenten von Wang [36]

Merkmale einer Tabelle

Inhalt

Der Inhalt einer Tabelle stellt eine Ansammlung von zusammenhängenden Begriffen bzw. Elementen dar. Diese Elemente können unter anderem Zahlen, reiner Text, Symbole, Figuren sowie mathematische Formeln sein. Wang bezeichnet diese Art der Daten als *entries*. Andere Elemente hingegen sind zusätzliche Daten, die die Funktion haben, die elementaren Daten in der Tabelle zu platzieren. Diese Daten werden als *labels* bezeichnet. Die *labels* werden in weitere Kategorien unterteilt. Diese Daten werden für die hierarchische Gliederung in einer Tabelle verwendet.
Abbildung 17 zeigt die durchschnittlich erreichten Punkte von unterschiedlichen Aufgaben und Prüfungen aus dem Jahr 1991 und 1992. Die Einträge, die die angegebenen Punkte zeigen, stellen die von Wang spezifizierten *entries* dar. Alle anderen Angaben, wie z.B. Jahr, bilden die *labels*. Diese enthalten weitere Kategorien, welche die Tabelle weiter unterteilt. Das Jahr ist zusätzlich in Jahreszeiten untergliedert. Die Aufgaben sind ebenfalls in Aufgabe eins bis drei unterteilt. Ein *label* kann aus weiteren *labels* bestehen. Zwischen den *entries* und *labels* herrschen logische Beziehungen. Jeder Eintrag wird z.B. mit einer Überschrift (*label*) aus jeder Kategorie assoziiert. Der erste Eintrag "85" in der Tabelle wird dem *label* 1991 sowie Winter und Ass1 zugeordnet. Die tabellarischen Elemente sowie die logischen Beziehungen untereinander, bestimmen die logische Struktur einer Tabelle. Die Zahl der vorhandenen Kategorien bestimmt die Größenordnung einer Tabelle. In der Tabelle aus Abbildung 17 sind drei Kategorien festzustellen und bildet somit eine drei-dimensionale Tabelle [11, 36, 37].

Darstellung der Form

Tabellen werden häufig in der Zeilen-Spalten Struktur dargestellt. Die Abbildung 17 ist in dieser Struktur angeordnet und zeigt die unterschiedlichen Komponenten einer Tabelle. Die Tabelle ist in vier Regionen gegliedert. Diese Regionen werden durch spezielle Grenzbereiche wie Abschnittsbegrenzung sowie Begrenzung des Tabellenkopfes identifiziert.
Die Einteilung der jeweiligen Abschnitte ist in der Abbildung 17 an der linken Seite zu finden. Dieser Bereich beinhaltet die Überschriften, welche die Zeilen beschreiben. Der Tabellenkopf enthält sämtliche Überschriften, um jede Spalte eindeutig identifizieren zu können. Der Tabellenkörper bildet den inneren Bereich einer Tabelle und ist unterhalb des Tabellenkopfes sowie auf der rechten Seite der definierten Abschnitte zu finden. Der Tabellenkörper enthält die Einträge der Tabelle. Der Schnittpunkt einer Zeile und einer Spalte bildet die Zelle, in welcher sich die von Wang spezifizierten *entries* befinden. *Labels* sind sowohl im Tabellenkopf, als auch in der Abschnittsdefinition der Tabelle zu finden. Spezielle Regeln sowie freie Räume, leere Zeilen und Spalten können genutzt werden, um Elemente voneinander abzugrenzen. Auch Hintergrundfarben sowie unterschiedliche Arten der Elemente können dazu beitragen, wichtige Informationen zu identifizieren [11, 36, 37].

3.4. Zusammenfassung

In diesem Kapitel wurden einleitend die grundlegenden Kenntnisse der Datenintegration dargestellt. Zu Beginn wurden die drei Grundprobleme, *Verteilung*, *Autonomie* und *Heterogenität* beschrieben. Weiterhin wurden Techniken zur Überwindung der Probleme vorgestellt. Gegenstand dieser Arbeit ist hauptsächlich die Überwindung der Heterogenität mittels des Schema Mapping Prozesses.
Im zweiten Teil des Kapitels wurden die Grundlagen für die Konzeption einer Datenbank beschrieben. Die Konzeption einer Datenbank ist ebenfalls Teilaufgabe der vorliegenden Arbeit und wird anhand des ER-Modells durchgeführt.
Da die zu integrierenden Datensätze in umfangreiche Excel-Tabellen gespeichert sind, wurden im dritten Abschnitt des Kapitels zwei Ansätze vorgestellt, mit deren Hilfe eine umfangreiche Analyse von Excel-Tabellen realisiert werden kann. Diese Ansätze werden in der Konzeption der Datenintegration verwendet.

4. Problemanalyse

Wie bereits in Kapitel 1 beschrieben, besteht das Ziel der vorliegenden Arbeit darin, ein Konzept zu entwickeln, um sämtliche Daten möglichst vollständig in die Datenbank integrieren zu können. Die Daten sind derzeit in umfangreiche Excel-Eingabeprotokolle der AG „Ökologie benthischer Organismen“ des IOW, gespeichert. Die Datenbestände der Excel-Eingabeprotokolle müssen dabei einheitlich, dauerhaft, effizient und widerspruchsfrei in eine Datenbank integriert werden.
In diesem Kapitel werden die Probleme bezüglich des Zusammenführens der unterschiedlich vorliegenden Excel-Eingabeprotokolle analysiert.

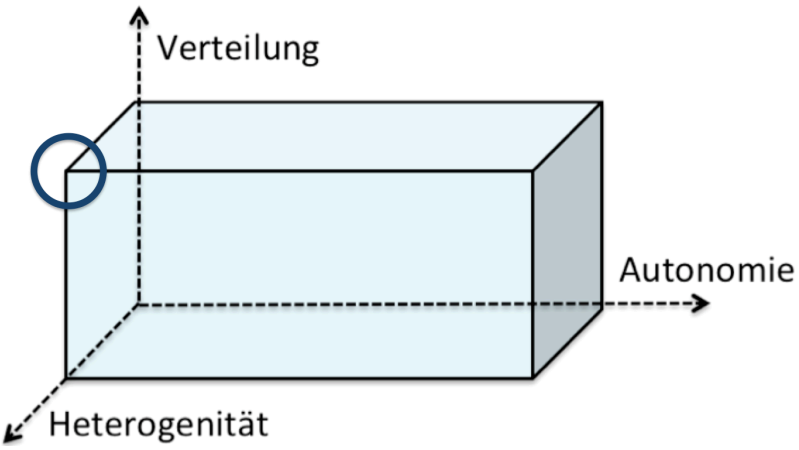


Abbildung 18: Einordnung der vorliegenden Datenquellen (Excel-Eingabeprotokolle) in die drei Dimensionen der Datenintegration

Die Abbildung 18 illustriert die bereits in Kapitel 3 vorgestellten Dimensionen der Datenintegration. Die zu integrierenden Excel-Eingabeprotokolle liegen als verteilte und heterogene Datenquellen vor. Die Datenquellen sind der Dimension *Verteilung* und *Heterogenität* zuzuordnen, siehe Abbildung 18. Das erste Hindernis, welches es zu überwinden gilt, liegt in der logischen Verteilung der Eingabeprotokolle. Die logische Verteilung wurde bereits in Kapitel 3.1.1 definiert. Jedes Eingabeprotokoll wird als eigenständige Excel-Datei angesehen. Die gesamten Datenbestände der Eingabeprotokolle liegen nicht in einer Excel-Datei vor, sondern sind auf etliche Dateien verteilt. Diese Dateien weisen sowohl inhaltliche, als auch strukturelle Unterschiede auf. Die Hauptprobleme in der Integration aller Excel-Eingabeprotokolle, treten bezüglich der Dimension *Heterogenität* auf.
Im weiteren Verlauf werden die aus 3.1.1 beschriebenen Grundprobleme der Datenintegration bezüglich der vorliegenden Excel-Eingabeprotokolle diskutiert. Dabei steht die Heterogenität der Datenquellen im Vordergrund.

In Abschnitt 4.1 wird sowohl eine inhaltliche-, als auch eine strukturelle Analyse der Eingabeprotokolle vorgenommen. Darauf aufbauend werden in Abschnitt 4.2 die auftretenden Probleme hinsichtlich der auftretenden Arten der Heterogenität untersucht und an konkreten Beispielen betrachtet. In diesem Zusammenhang werden abschließend in Abschnitt 4.3 die in diesem Kapitel gewonnenen Erkenntnisse zusammengefasst und Lösungsansätze aufgelistet, die konkret in Kapitel 5 vorgestellt werden.

4.1. Analyse der Quelldaten

Im ersten Teil dieses Abschnittes wird eine inhaltliche und im zweiten Teil eine strukturelle Analyse der vorliegenden Excel-Eingabeprotokolle vorgenommen.

4.1.1. Inhaltliche Analyse

In Abschnitt 2.2 wurden bereits die aufwändige Gewinnung der Benthosproben vorgestellt. Dabei wurden ausführlich die zu untersuchenden Daten wie z.B. die Trocken- (TM), Feucht- (FM) sowie aschefreie Trockenmasse (AFTM) beschrieben. Die zur Verfügung stehenden Excel-Eingabeprotokolle speichern unter anderem Metadaten pro Fund sowie die zu untersuchenden Daten der benthischen Organismen und deren gewonnenen Ergebnisse und belaufen sich auf ca. 200 000 Datensätze.
In der Abbildung 19 werden die gespeicherten Informationen der einzelnen Eingabeprotokolle, geordnet nach Jahreszahl, zusammengefasst. Es werden jeweils nur neu protokollierte Informationen, ausgehend vom ersten zur Verfügung stehenden Excel-Eingabeprotokoll von 1998, in der zweiten und dritten Spalte aufgelistet. Die zweite Spalte bezieht sich auf die Metadaten pro Fund und die dritte Spalte auf die spezifischen Informationen der erhobenen Benthos-Organismen, von den Greifer-Proben 1 bis 3 und der Dredge. Zusätzlich wird in dieser Spalte auch die Informationen zur Auswertung aufgelistet.

	Excel-Eingabeprotokolle	
Jahr	**Zusätzlich protokollierte Matadaten pro Jahr**	**Zusätzlich protokollierte Benthosdaten pro Jahr**
1998	Station, Datum, Koordinaten, Salzgehalt	Taxa, Anzahl
1999 - 2001	Schiff, Tiefe, Gerät, Probenanzahl, Fläche, Siebweite	- Probennummer (Greifer 1 bis 3): Taxa, Anzahl, FG, TG, AFTG, Faktoren - Dredgehol: Taxa, Anzahl - Protokollauswertung, berechnet: Taxa, Abundanz, FG, TG, AFTG
2002	identisch	identisch
2003	identisch	identisch
2004	identisch	- Dredgehol: Taxa + "gefunden in" -> Hol 1 - 3 & Dredge
2005/06	identisch zu 1998	identisch zu 1998
2007	identisch zu 2003	identisch zu 2003
2008	identisch	- Pobennummer (Greifer 1 - 3): zusätzlich jeweils Tara-Gewicht: T+FG, T+TG, T+AFTG - Protokollauswertung, berechnet: zusätzlich: Bearbeiter Labor
2009	identisch	identisch
2010	- Projekt: Bezeichnung, ID, Auftraggeber, Ansprechpartner, ... - Station: Zeit, Salzgehalt, Sauerstoff, Temperatur, Seegebiet, Video, Sonstiges - Angaben zu Hol (Greifer + Dredge): Bezeichnung, HolNr., Sediment-ansprache, Bearbeiter Labor, Wäger_FM + TM + AFTM, Datum, QKEingabe, Gerät, Fläche	- Probennummer (Greifer/Hol 1 - 3): Eingabe d. Daten - Dredgehol: Eingabe d. Daten - Kalkulator: Berechnungen - Artenliste: Nr., Artname, Gattung, Art, ... - Faktoren Ostsee: Art, Anzahl, FM zu TM, FM zu AFTM - Ergebnisprotokoll (Hol 1 - 3 & Dredge)
2011/12	- Station: Schiff	- Stationsprotokoll - Gruppen_Rang
2013	- Angaben zu Hol (Greifer + Dredge + Probe "Sediment" + zu jeder Probe Positionsangabe	identisch

Abbildung 19: Übersicht der jeweils zusätzlich protokollierten Informationen der Eingabeprotokolle, aufsteigend nach Jahreszahl

Anhand der Abbildung 19, die als inhaltliche Übersicht dienen soll, werden im Folgenden die inhaltlichen Unterschiede der Excel-Eingabeprotokolle erläutert.
Das älteste Excel-Eingabeprotokoll von 1998 beinhaltet, im Gegensatz zum neuste Excel-Eingabeprotokoll von 2013, eine geringe Menge an Informationen. Relevant sind hierbei Daten, die sich auf die angefahrenen Stationen beziehen. Dazu zählen Informationen wie Stationsname, Datum und Koordinaten der Station. Weiterhin werden in dem Eingabeprotokoll anschließend die Taxa und Anzahl der gefundenen Taxa aufgelistet.
Werden die Excel-Eingabeprotokolle von 1998 und 1999 bis 2001 verglichen, so sind deutlich mehr Informationen protokolliert. Zusätzlich zu den bereits protokollierten Daten werden Wassertiefe, aber auch andere allgemeine Informationen wie Schiff, Anzahl der gewonnenen Proben sowie Name der verwendeten Geräte und deren Fläche und Siebweite gespeichert und weiterhin die einzelnen Proben aufgelistet. Darunter zählen die Greifer-Proben 1 bis 3 und die Dredge. Die erhobenen Proben, sowie deren Definition, wurde bereits in Kapitel 2 beschrieben. Hauptbestandteil der aufgelisteten Proben 1 bis 3 und der Dredge sind die gefundenen Taxa. Welche Informationen konkret zu den Proben gespeichert werden, sind der Abbildung 19 zu entnehmen. Die Definitionen der Abkürzungen wurden ebenfalls in Kapitel 2 geklärt. Das Eingabeprotokoll speichert zusätzlich eine generierte Protokollauswertung aus berechneten Werten.
Die Excel-Eingabeprotokollen von 2002 und 2003 sind identisch. Das Eingabeprotokoll von 2004 weist einen Unterschied in der Auflistung der Daten des Dredgehols auf. Hierbei werden die gefunden Taxa in "gefunden in" eingeteilt und jeweils die Probennummer der Greifer 1 bis 3 angegeben. An dieser Stelle sei zu erwähnen, dass jede Art von Probennahme *Hol* genannt wird. Ab dem Eingabeprotokoll von 2004 werden die Greifer-Proben 1 bis 3 als Hol 1 bis 3 bezeichnet.
Die Unterschiede zwischen den Excel-Eingabeprotokollen von 2005 bis 2007 sind der Abbildung 19 zu entnehmen.
Ab dem Eingabeprotokoll von 2008 werden zu den Proben 1 bis 3 jeweils die Tara-Gewichte gespeichert. Bis einschließlich dem Jahr 2009 sind die Eingabeprotokolle ansonsten weitestgehend identisch.
In den neueren Eingabeprotokollen, ab 2010, werden deutlich mehr Daten erfasst. So wird eine Einteilung der protokollierten Daten in *Projektinformationen*, *Stationsinformationen* und *Angaben zu den Hols* vorgenommen. Zusätzlich werden unter anderem Daten wie Zeit, Salzgehalt, Sauerstoff, Temperatur und Seegebiet gespeichert. Bei den Angaben zu den Hols erfolgt eine weitere Einteilung der erhobenen Proben in Hol 1 bis 3 sowie der Dredge. Zu jedem Hol werden weitere spezifische Daten gespeichert. Unter anderem werden vier unterschiedliche Bearbeitungstypen mit jeweils einem Datum, an dem die Probe bearbeitet wurde, gespeichert. Dazu zählen Bearbeiter Labor, Wäger FM, Wäger TM und Wäger AFTM. Eine weitere Person wird für die Eingabe der Qualitätskontrolle angegeben.
Auffällig bei den Excel-Eingabeprotokollen von 2011/12 ist die Anzahl der beinhalteten Tabellenblätter im Excel-Dokument. Alle Informationen zu den gefundenen Benthos-Arten jedes Hols und der Dredge, sind jeweils in eigenständige Tabellenblättern vorhanden und beinhalten die gleichen Informationen, die in den älteren Eingabeprotokollen von 2010 aufgelistet sind.

Weitere Tabellenblätter speichern jeweils eine Artenliste, die Umrechnungsfaktoren für die Ostsee sowie Ergebnisprotokolle. Die Ergebenisprotokolle werden jeweils aus den bereits protokollierten Information der Hols und der Dredge generiert.
Die neusten Eingabeprotokolle von 2013 speichern neue Informationen in Bezug auf die Angaben der Hols. Hinzu kommt eine weitere Probe namens *Sediment*. Zu jeder der Proben werden zusätzlich die exakten Koordinaten (Positionierung) gespeichert.
In den Excel-Eingabeprotokollen ist, wie bereits erwähnt, jeweils eine generierte Auswertung gespeichert. Die in der Auswertung aufgelisteten Werte sind berechnete Werte aus Eingabedaten der Benthos-Proben.
Anhand der inhaltlichen Analyse ist ein deutlicher Anstieg der protokollierten Informationsmenge über das Vorkommen von Benthos-Organismen sowie weiteren Parametern zu erkennen. In diesem Zusammenhang entwickelten sich strukturell regelmäßig neue Excel-Eingabeprotokolle, die wiederum jeweils einem Excel-Protokolltyp zugeordnet sind. Im weiteren Verlauf der Arbeit wird der Begriff *Excel-Protokolltyp* verwendet. Dieser beschreibt eine Klasse von Excel-Dokumenten, die die gleiche Struktur aufweisen und zu einem bestimmten Zeitpunkt in einer eindeutigen Struktur vorliegt.

4.1.2. Strukturelle Analyse

In diesem Abschnitt werden die Strukturunterschiede der Excel-Eingabeprotokolle analysiert und gleichzeitig die Excel-Protokolltypen abgeleitet. Daraus ergibt sich die Anzahl der vorliegenden Protokolltypen.
Nachdem die Daten der erhobenen Benthos-Proben auf Papier erfasst worden sind, werden diese anschließend für eine digitale Auswertung strukturiert in Excel-Tabellen übertragen und gespeichert. Wie bereits erwähnt, liegen die Datenbestände von 1998 bis heute in einer historisch gewachsenen Struktur vor.
Vorab ist zu erwähnen, dass aus der durchgeführten Strukturanalyse hervor geht, dass sich zehn unterschiedliche Excel-Protokolltypen herausgebildet haben, die im Folgendem anhand ihrer Struktur beschrieben werden.

Protokolltyp Nr. 1
Der Aufbau eines Excel-Tabellenblattes wurde bereits in Kapitel 3.3 beschrieben. In Excel können in einer Arbeitsmappe eine Reihe von Tabellenblätter enthalten sein.
Der erste Protokolltyp von 1998 beinhaltet ein Tabellenblatt, welches die gesamten Informationen auflistet. In der ersten Zelle (Zeile 1, Spalte A) ist eine Überschrift protokolliert. Bei diesem Protokolltyp sind lediglich die ersten beiden Spalten (Spalte A und B) mit Informationen belegt. Ab Zeile 12 beginnt eine definierte Tabelle mit einer einfachen Struktur. Diese Tabelle enthält zwei Spalten. Die Zeilenanzahl kann von Protokoll zu Protokoll, bzw. von Benthosprobe zu Benthosprobe variabel sein. In dieser Tabelle werden alle in einer Probe gefundenen Benthos-Arten aufgelistet.

Protokolltyp Nr. 2

Der zweite Protokolltyp von 1999 bietet die Möglichkeit, mehr Informationen einzutragen. In der ersten Zelle ist ebenfalls eine Überschrift für das Eingabeprotokoll eingetragen. Anschließend folgen leere Zeilen, um die für die Untersuchung spezifische Informationen zu kennzeichnen. Ein wesentlicher unterschied liegt in der Belegung der Spalten. Die protokollierten Informationen erstrecken sich über weitere Spalten. Ein weiterer Unterschied liegt in der Einteilung der protokollierten Benthosproben. Diese werden in dem vorliegendem Protokolltyp mit der Bezeichnung *Probennummer 1* bis *Probennummer 3* sowie *Dredgehol* aufgelistet. Die dazugehörigen Informationen werden in eine separate Tabelle, mit jeweils acht Spalten und einer variierenden Zeilenanzahl, untereinander eingetragen. Insgesamt können bis zu vier Tabellen untereinander aufgelistet werden. Jede Tabelle ist mit einer Überschrift, Angabe der Probennummer und weiteren Überschriften zu den jeweiligen acht Spalten gekennzeichnet. Die Tabelle, in der die Informationen zum Dredgehol aufgelistet sind, beinhaltet lediglich 2 Spalten und gleicht somit dem ersten Protokolltyp.
Nach anschließender Auflistung der vier Tabellen ist eine weitere Tabelle zu erkennen. Diese letzte Tabelle beinhaltet zum einen Kopien der bereits weiter oben aufgelisteten Informationen und zum anderen Angaben zur Abundanz, zum Feucht-, Trocken- und aschefreien Trockengewicht zu den jeweiligen Benthos-Arten. Diese Tabelle hat einen deutlich erkennbaren Rahmen und beinhaltet 16 Spalten, mit spezifischen Überschriften, in denen die bereits erwähnten Informationen aufgelistet sind.
Aus der Analyse geht hervor, dass es sich um einen weiteren Protokolltyp handelt, der eine umfangreichere Struktur als sein Vorgänger aufweist. Aufgrund dieser Veränderung kann darauf geschlossen werden, dass sich die Anforderungen verändert haben und mehr Informationen protokolliert werden mussten, die in das Protokoll aufzunehmen waren.

Protokolltyp Nr. 3

Die Zeilenanzahl kann von Eingabeprotokoll zu Eingabeprotokoll variieren. Dies ist auf die Zahl der gefunden Taxa zurückzuführen. Entscheidend für die Analyse ist die Anzahl der belegten Spalten. Ein wesentlicher Unterschied zum vorherigem Protokolltyp ist, dass die Tabellen, die die Informationen zu den Benthosproben auflisten, sieben statt acht Spalten aufweisen. Beim vorherigem Protokolltyp wurde die sechste Spalte frei gelassen, diese ist bei dem vorliegenden Protokolltyp mit Informationen belegt. Dieser Unterschied spielt bei der Adressierung der Werte für die spätere Integration eine entscheidende Rolle. Hierbei muss genau festgelegt werden, an welcher Position im Excel-Eingabeprotokoll welche Daten zu finden sind. Weiterhin sind keine strukturellen Unterschiede zum vorherigem Protokolltyp festzustellen.

Protokolltyp Nr. 4

Bei diesem Protokolltyp ab 2004 findet sich ein offensichtlich struktureller Unterschied in der Darstellung der Tabelle des Dredgehols wieder. Die Spaltenanzahl hat sich von zwei auf vier erhöht. Jeder Spalte ist mit einer Überschrift gekennzeichnet.
Weiterhin ist zu erwähnen, dass nicht nur die Zeilenanzahl, sondern auch die Anzahl der Tabellen zu den einzelnen Benthosproben variieren kann. Es kann durchaus der Fall ein-

treten, dass nur eine oder zwei Benthosproben aufgelistet werden. Ein weiterer Unterschied findet sich nach der Auflistung des Dredgehols in der letzten Tabelle wieder. An dieser Position sind zwei weitere Belegungen von Zellen festzustellen, die entscheidende Informationen beinhalten könnten und somit nicht außer Acht gelassen werden dürfen.

Protokolltyp Nr. 5
Aus dem Jahr 2005 und 2006 sind Eingabeprotokolle vorhanden, die strukturell dem Protokolltyp Nr. 1 ähneln. Der einzige Unterschied liegt in einer neu belegten Spalte, die durch eine Überschrift gekennzeichnet ist. Durch die zusätzliche Spalte werden auch dieses Eingabeprotokolle als neuer Protokolltyp betrachtet.

Protokolltyp Nr. 6
Die Eingabeprotokolle ab 2007 weisen eine komplett neue Struktur auf. Die Informationen zu den einzelnen Benthosproben sind zusammen in einer Tabelle aufgelistet, und nicht wie zuvor in eigenständige Tabellen zeilenweise untereinander. Die neue Struktur ist in Abbildung 20 dargestellt. Auf der linken Seite sind die bisher in separate Tabellen aufgelisteten Probennummern abgebildet und auf der rechten Seite die neue Tabelle. Diese beinhaltet 15 Spalten. Die einzelnen Probennummern sind spaltenweise nebeneinander angeordnet.
Die erste Spalte der Tabelle listet die Namen der Benthos-Arten auf. Die Probennummern sind im Tabellenkopf der jeweils darauf folgenden Spalten eingetragen Jede Probennummer hat weitere Untergruppen, die ebenfalls im Tabellenkopf eingetragen sind. Unter dem Tabellenkopf sind sämtliche Informationen zu den Benthos-Proben aufgelistet. Ein weiterer Unterschied ist in der Angabe des Dredgehols zu finden. Die Tabelle des Dredgehols ist in der Struktur und der Anzahl der Spalten, identisch mit der Tabelle des Dredgehols aus dem Protokolltyp Nr. 1.

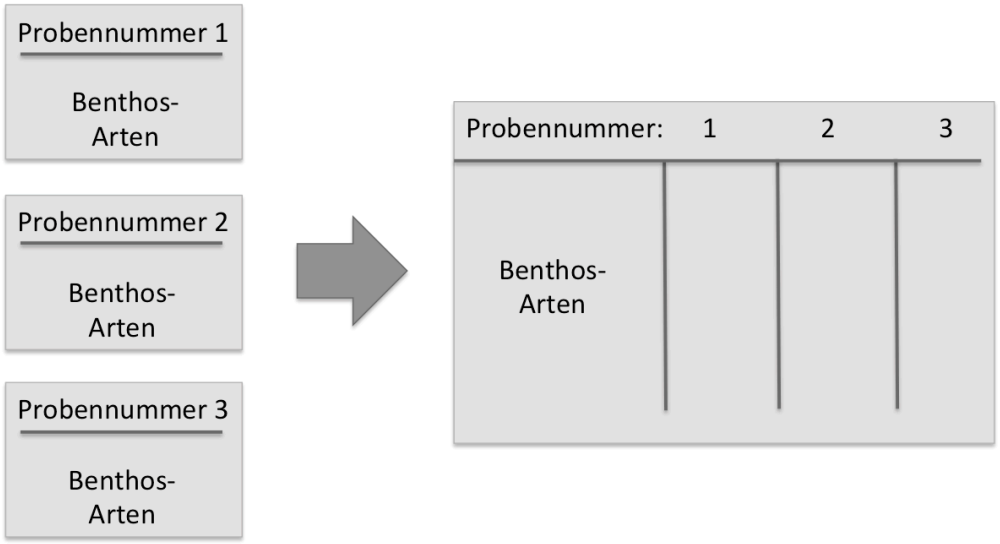


Abbildung 20: Auflistung der Benthosproben, links Protokolltyp Nr. 2 bis 5, rechts Protokolltyp Nr. 6

Protokolltyp Nr. 7

Die Eingabeprotokolle von 2008 und 2009 ähneln strukturell dem Protokolltyp Nr. 4. Die Proben werden jeweils wieder in separate Tabelle untereinander angeordnet. Die Spaltenanzahl zu jeder Probe beträgt neun. Jede Tabelle sowie jede dazugehörige Spalte, hat eine Überschrift. Ein Unterschied zum Protokolltyp Nr. 4 liegt in der letzten Tabelle. Diese beinhaltet weitaus mehr belegte Zellen. Ein weiterer Unterschied liegt in der Tabelle zum Dredgehol. Die Spaltenanzahl hat sich von vier auf zwei verringert.

Protokolltyp Nr. 8

Einen Umbruch in der Protokollstruktur existiert seit 2010. Jedes Excel-Dokument enthält eine Reihe an Tabellenblätter. Alle zuvor beschriebenen Protokolltypen beinhalteten sämtliche Informationen auf einem Tabellenblatt. Der vorliegende Protokolltyp beinhaltet zwölf Tabellenblätter, die jeweils sowohl inhaltliche-, als auch strukturelle Unterschiede aufweisen. Jedes Tabellenblatt ist mit einer eindeutigen Bezeichnung gekennzeichnet.
Das erste Tabellenblatt, in Abbildung 21 dargestellt, listet eine Reihe an Informationen auf, die strukturiert in drei voneinander abgrenzenden Abschnitten eingeteilt sind. Diese Abschnitte können als eigenständige Tabellen betrachtet werden. Die Tabellen sind durch einen äußeren Rahmen und durch eine Hintergrundfarbe gekennzeichnet.

	A	B	C	D	E	F	G	H
1	**Projektbezeichnung:**		Ostseemonitoring 2010			**ID:**	Arbeitsplan 2010 des IOW für das BSH	
2	**Auftraggeber:**		BSH			**Ansprechpartner:**		
3	**gültige Artenliste:**		Artenliste Version 2.00, Stand 18.02.2011				Dr. Marion Heinrich	
4	**gültige Liste Bestimmungsliteratur:**		Literaturliste Version 1.00, Stand 09.11.2010					
5								
6	**Angaben zur Station**							
7								
8	**Stationsname:**	010 (N1)			**Seegebiet:**		Fehmarnbelt	
9	**Datum Probenahme:**	9.11.2010						
10	**Zeit (lokal):**	16:30			**Positionierung (WGS 84):**			
11	**Maschenweite:**	1 mm			**E°min,dec**		11,312	
12					**N°min,dec**		54,553	
13	**Wassertiefe (m):**	27,56						
14	**Salzgehalt (ppt):**	22,9			**Zusätzliche Untersuchungen:**			
15	**Sauerstoff (mg/l):**	5,9			**Video:**			
16	**Temperatur (°C):**	10,0						
17					**Sedimente:**		X	
18	**Zahl quantitative Proben:**	3						
19	**Angaben zur Biomasse:**	FM, TM und AFTM gemessen			**Sonstige:**			
20								
21	**Angaben zu den Hols**							
22								
23	**Probenbezeichnung**	**010 (N1)_1_9.11.2010**		**010 (N1)_2_9.11.2010**		**010 (N1)_3_9.11.2010**		**010 (N1)_D_9.11.2010**
24	**Hol**	1		2		3		D
25	**Sedimentansprache Bord**	Feinsand mit geringem Schlickanteil		Feinsand mit geringem Schlickanteil		Feinsand mit geringem Schlickanteil		
26	**Bearbeiter Labor**	I. Glockzin		I. Glockzin		I. Glockzin		N. Keiser
27	**Datum Laborbearbeitung**	16.02.2011		17.02.2011		16.02.2011		6.1.2011
28	**Wäger_FM**	I. Glockzin		I. Glockzin		I. Glockzin		
29	**Datum Wägung_FM**	16.02.2011		17.02.2011		16.02.2011		
30	**Wäger_TM**	I. Glockzin		I. Glockzin		I. Glockzin		
31	**Datum Wägung TM**	17.02.2011		18.02.2011		17.02.2011		
32	**Wäger_AFTM**	I. Glockzin		I. Glockzin		I. Glockzin		
33	**Datum Wäger_AFTM**	18.02.2011		19.02.2011		18.02.2011		
34	**QK Eingabe**	A. Zettler		A. Zettler		A. Zettler		A. Zettler
35	**Gerät**	van Veen		van Veen		van Veen		Dredge
36	**Fläche/ Hol (m²)**	0,0986		0,0986		0,0986		0

Abbildung 21: Struktur des Tabellenblattes Nr. 1; Protokolltyp Nr. 8

Die Informationen der jeweiligen Benthos-Proben sind in darauffolgende Tabellenblätter protokolliert. Für jede Benthosprobe ist jeweils ein eigenes Tabellenblatt reserviert, die 15 gekennzeichnete Spalten beinhalten. Die Zeilenanzahl kann von Protokoll zu Protokoll variieren.

Protokolltyp Nr. 9

Der Protokolltyp ab 2011 beinhaltet 15 Tabellenblätter, statt 12 beim vorherigen Protokolltyp. Strukturell sind die beiden Protokolltypen gleich. Der einzige Unterschied ist im zweiten Abschnitt des ersten Tabellenblattes zu finden. In diesem Abschnitt sind zwei belegte Zellen hinzu gekommen.

Protokolltyp Nr. 10

Der aktuellste Protokolltyp von 2013 ist ebenfalls bezüglich der Struktur mit dem Protokolltyp von 2010 und 2011 identisch. Der einzige Unterschied ist im ersten Tabellenblatt festzustellen. Im dritten Abschnitt ist, wie in Abbildung 22 zu erkennen, eine weitere Spalte sowie zwei Zeilen hinzugekommen.

Probenbezeichnung Hol	FBR07_1_05.06.2013 1	FBR07_2_05.06.2013 2	FBR07_3_05.06.2013 3	FBR07_D_05.06.2013 D	FBR07_Sediment_05.06.2013 Sediment
Sedimentansprache Bord	etwas Kies, Mergel	dünne Kiesauflage auf grauem Mergel	Grobsand, Kies, etwas Mergel		
Bearbeiter Labor	N. Keiser	J. Harder/I. Glockzin	N. Keiser	I. Glockzin	Bearbeiter Korngröße:
Datum Laborbearbeitung	28.08.2013	30.08.2013	29./30.08.2013	29.08.2013	J. Harder
Wäger_FM	N. Keiser	I. Glockzin	N. Keiser		CILAS/Siebturm:
Datum Wägung_FM	28.08.2013	30.08.2013	30.08.2013		Siebturm
Wäger_TM					Bearbeiter Organik:
Datum Wägung TM					I. Glockzin
Wäger_AFTM					
Datum Wäger_AFTM					
QK Eingabe	A. Zettler	A. Zettler	A. Zettler	A. Zettler	
Gerät	van Veen	van Veen	van Veen	Dredge	
Fläche/ Hol	0,1023	0,1023	0,1023	0	-
Position E°min,dec	11°00,5805	11°00,5897	11°00,6163	11°00,6938	
Position N°min,dec	54°37,6656	54°37,6675	54°37,6706	54°37,6747	

Abbildung 22: Dritter Abschnitt des ersten Tabellenblattes; Protokolltyp Nr. 10

In den vorliegenden Excel-Eingabeprotokollen sind Unterschiede in der Struktur festzustellen. In diesem Anschnitt wurden die Unterschiede beschrieben. Anhand der strukturellen Analyse konnten zehn unterschiedliche Excel-Protokolltypen abgeleitet werden, die jeweils eine eigene Struktur aufweisen.

4.2. Heterogenität der Quelldaten

In den vorherigen Anschnitten wurde eine inhaltliche- und strukturelle Analyse der Excel-Eingabeprotokolle durchgeführt. Dabei wurden vor allem die Unterschiede zwischen den Eingabeprotokollen verdeutlicht.
In diesem Anschnitt werden die konkreten Probleme bezüglich der *Heterogenität* der Quelldaten beschrieben. Die Unterschiede in den vorliegenden Quelldaten werden im Folgenden anhand der fünf auftretenden Arten der Heterogenität beschrieben. Hierbei werden konkrete Beispiele aus den Excel-Eingabeprotokollen erläutert.
In der Tabelle 2 werden die Unterschiede vorab zusammengefasst. Die Tabelle listet die anhand der Protokolltypen auftretenden Arten von Heterogenität auf. Die Definitionen der unterschiedlichen Arten von Heterogenität sind der Tabelle 1 aus dem Kapitel Grundlagen zu entnehmen.

Heterogenität	**Eigenschaft**	**Protokolltyp**	**Repräsentation**
Technische	nicht vorhanden	-	-
Syntaktische	Flächenangabe	1 - 7 8 - 10	cm^2 m^2
	Siebweite Maschenweite	1 - 7 8 - 10	μm mm
Datenmodell	Datei-Format	variabel	xls.; xlsx.; xlsm.
Strukturelle	Darstellung der Proben	1 2 - 5 6 - 7 8 - 10	Anordnung der Informationen
Semantische	Gleiche Intension	1 - 7 8 - 10	Probennummer Hol
		1 -7 8 - 10	Siebweite Maschenweite
		1 - 7 8 - 10	Koordinaten Positionierung in E & N
		1 - 7 8 - 10	Dredgehol Dredge

Tabelle 2: Zusammenfassung der Arten von Heterogenität bezüglich der Protokolltypen

Bezüglich der Quelldaten ist keine **Technische Heterogenität** festzustellen, da der Zugriff auf die Eingabeprotokolle problemlos erfolgen kann.
Ein Schwerpunkte stellt die Probleme der **syntaktischen Heterogenität** dar. Einige der zu integrierenden Informationen werden zwischen den Protokolltypen auf unterschiedliche Art und Weise dargestellt. So werden die syntaktischen Unterschiede in der Darstellung der zu protokollierenden Flächenangabe der Greifer-Proben deutlich. In den älteren Protokolltypen wird unter anderem die Flächenangabe in cm^2, in den neueren Protokolltypen in m^2 angegeben. Weiterhin unterscheidet sich die Darstellung der Sieb- bzw. Maschenweite zwischen den Protokolltypen. Zum einen wird die Weite in μm und zum anderen in mm angegeben. Bei der Integration muss darauf geachtet werden, dass sämtliche Angaben aus den Protokollen in ein einheitliches Format umgerechnet und in der Datenbank abgelegt werden. Ein weiteres Beispiel der syntaktischen Heterogenität findet sich zusätzlich in der Darstellung des Stationswertes. Hierbei werden sowohl Kürzel aus Buchstaben und Zahlen, als auch konkrete Namen angegeben. Dieser Wert sollte im Schema so modelliert werden, dass hierfür ein bestimmter Datentyp festgelegt wird.
Zusätzlich ist zwischen den Eingabeprotokollen die **Datenmodellheterogenität** festzustellen, da die Excel- Eingabeprotokolle in unterschiedlichen Excel-Dateiformaten vorliegen.

1. **XLS**
   Excel 97 - Excel 2003, basiert auf einem offenen Binärdateiformat von Microsoft

2. **XLSX**
   Extensible Markup Language (XML)-basiertes Office Excel 2007 Standarddateiformat, kann weder Makro noch Visual Basic for Application (VBA) speichern

3. **XLSM**
   XML-basiertes Office Excel 2007 Dateiformat mit Makros

Die Excel-Eingabeprotokolle von 1998 bis 2008 liegen im *.xls*-Dateiformat vor. Die neueren Protokolle ab 2009 bis heute sind im *.xlsx*- oder im *.xlsm*-Format gespeichert. Diese Art von Heterogenität gilt es ebenfalls zu überwinden. Ein Lösungsansatz ist die Überführung aller Excel-Dateien in ein einheitliches frei wählbares Dateiformat. Dieser Ansatz wird in Kapitel 5 vorgestellt.
Strukturelle Heterogenität wurde bereits durch die zuvor durchgeführte Strukturanalyse ersichtlich. Hierbei treten Probleme bezüglich der strukturellen Darbietung der protokollierten Daten auf. Die strukturellen Unterschiede werden bereits beim Vergleich des ersten und zweiten Protokolltyps deutlich. Die Unterschiede liegen in der bereits erwähnten Repräsentation der aufgelisteten Benthos-Proben. Die strukturellen Unterschiede ziehen sich dabei durch sämtliche Protokolltypen hindurch. Die strukturellen Probleme können zudem durch eine vereinheitlichte Darstellung gleicher Informationen beseitigt werden. Die Vereinheitlichung soll im weiteren Verlauf durch die Konzeption eines einheitlichen Zielschemas in Kapitel 5 durchgeführt werden.
Die *semantische Heterogenität* bezieht sich auf die Probleme bezüglich der Bedeutung von Begrifflichkeiten. Ein semantischer Konflikt ist zwischen den Protokolltypen 1 bis 7 und 8 bis 10 zu erkennen. Hierbei wird sowohl der Begriff *Probennummer*, im Protokolltyp Nr. 1 bis 7, als auch *Hol*, in Protokolltyp Nr. 8 bis 10, verwendet. Diese beiden Begrifflichkeiten sind

Synonyme, die jeweils in den Protokolltypen die selbe Bedeutung haben. Bei der Integration muss darauf geachtet werden, dass die richtigen Daten ausgewählt werden und dass sowohl das Zielsystem, als auch die in den Eingabeprotokollen vorliegenden Datensätze dasselbe meinen bzw. die gleiche Bedeutung haben [14, 24, 33].

Um die auftretenden Probleme bezüglich der Heterogenität zu lösen, wurde bereits Lösungsvorschläge aufgelistet. Zum Einsatz kommen hierbei vor allem die aus 3 beschriebenen Techniken der Informationsintegration, die Ulf Leser und Felix Naumann in [24] beschreiben. Der Schwerpunkt liegt dabei in den zu überwindenden semantischen und strukturellen Unterschieden. Nachdem die Probleme überwunden sind, können die Datenbestände in einem globalen Zusammenhang betrachtet und in das Zielsystem integriert werden [24, S. 317f].

4.3. Zusammenfassung

In diesem Kapitel wurden die Probleme bezüglich der Integration möglichst aller Excel-Eingabeprotokolle diskutiert und vor allem die Probleme der Heterogenität in den Vordergrund gestellt. Diese Probleme können mit bestimmten Techniken und Verfahren überwunden werden. Im folgenden Kapitel werden konkrete Lösungsansätze vorgestellt.
In diesem Zusammenhang ist eine exakte Analyse der vorliegenden Excel-Eingabeprotokolle bezüglich Inhalt und Struktur notwendig. Werden fehlerhafte Daten im Zielsystem gespeichert, so kann keine korrekte Auswertung aus den über die Jahre aus Forschungen erhobenen Daten, sichergestellt werden. Die Folge sind falsche Ergebnisse.
Die Excel-Eingabeprotokolle weisen verschiedene Attribute[29] und Schemata[30] auf. Eine solche verteilte Datenhaltung kann negative Folgen haben. Dabei können Redundanzen sowie Widersprüche, z.B. Dopplungen, in den Datensätzen auftreten. Werden gleiche Daten verteilt gespeichert, dann können z.B. beim Bearbeiten von Informationen Widersprüche auftreten. Diese Probleme gilt es, durch eine homogene Darstellung der Informationen zu beheben [8, 23, 24]. Das heißt, dass aus sämtlichen Schemata ein einheitliches Schema abgeleitet werden muss. Eine konkrete Lösung wird in Kapitel 5 vorgestellt.
Leser und Naumann spezifizieren in [24, S. 61] Methoden zur Überwindung der Heterogenität. Dabei beschreiben sie, dass die Heterogenität erst überwunden sei, wenn die Probleme der auftretenden Arten der Heterogenität beseitigt sind. Zu diesem Zweck müssen unter anderem die Elemente des Zielsystem sowie der vorliegenden Quelldaten die gleiche Bedeutung aufweisen. Die in den Excel-Eingabeprotokollen vorliegenden Daten müssen auch als solche im Zielsystem modelliert werden sowie alle vorhandenen Informationen im Zielsystem einheitlich dargestellt werden. Das Datum sowie die Zeit müssen auch als solche, im selben Format, im Zielsystem auftauchen. Sind in den Quelldaten unterschiedliche Zeit- und Datumsangaben zu finden, so müssen diese Angaben in ein einheitliches Format transformiert werden. Außerdem müssen semantisch zusammenhängende Konzepte auch in ihrer Struktur einheitlich modelliert werden.

[29] Ein Attribut ist ein Merkmal bzw. eine Eigenschaft.
[30] Das Schema ist im Allgemeinen die Darstellung eines Sachverhaltes bzw. in der Informatik ein Modell, um die Struktur von Daten zu repräsentieren.

Die vorliegenden Datensätze unterscheiden sich weiterhin in Eigenschaften wie Aktualität, Detailliertheit und Genauigkeit. Diese Eigenschaften müssen verlässlich analysiert werden. Die Aktualität der zu protokollierenden Daten stellt hierbei eine weitere Herausforderung dar. Dabei sind in einem älteren Excel-Eingabeprotokoll zu einer bestimmten Zeit die aktuellen Namen der Benthos-Arten eingetragen. Hat sich jedoch die Taxonomie sowie Namen der Benthos-Arten verändert, so sind zu einem anderem Zeitpunkt die Namen der Benthos-Arten des älteren Protokolls nicht mehr aktuell.
Hierbei stellt sich die Frage, wie dieser Sachverhalt am geeignetsten modelliert werden kann, sodass nicht nur die aktuelle Taxonomie in der Datenbank gespeichert ist. Dies ist notwendig, um korrekte Ausgabeprotokolle auch für ältere Untersuchungen generieren zu können, in der die zu einer bestimmten Zeit aktuellen Namen der Benthos-Arten aufgelistet werden können. Ein Ansatz ist bspw. eine automatische Verlinkung zu einer anderen Datenbank, mit deren Hilfe die Taxonomie der letzten Jahre nachweisbar ist. Dies ermöglicht, sämtliche geänderte Namen der Benthos-Arten abzurufen und Informationen zu Autor und Entdeckungsjahr zu erhalten. Diese Idee wird in der vorliegenden Arbeit jedoch nicht konkret umgesetzt. Dieser Ansatz sollte für eine Weiterentwicklung der Datenbank nicht außer Acht gelassen werden.
Ein weiterer Punkt ist die Adressierung der zu integrierenden Daten in den Eingabeprotokollen. Diese Daten müssen in den unterschiedlichen Excel-Protokolltypen identifizieren werden. Da in jedem der zehn Protokolltypen die Informationen an unterschiedlichen Position in den Eingabeprotokollen zu finden sind, muss daher explizit die Position spezifiziert werden. Auch für dieses Problem muss ein Lösungsansatz gefunden werden, um die Werte aus der Excel-Datei auslesen zu können und anschließend für die Integration zur Verfügung zu stellen. Dieser Lösungsansatz wird in Abschnitt 5.2.2 vorgestellt.

5. Konzeption

Dieses Kapitel beschreibt ein Konzept, mit deren Hilfe die Integration möglichst aller relevanten Quelldaten, aus den unterschiedlichen Excel-Protokollen, in ein globales Zielsystem durchgeführt werden kann. Das Kapitel ist in zwei Abschnitte aufgeteilt. Der erste Teil, 5.1, beschäftigt sich mit der Thematik der Datenbankmodellierung, der zweite Teil 5.2 mit der eigentlichen Datenintegration.
Das Zielsystem hat die Aufgabe, die Daten strukturiert aufzubewahren und zuverlässig zur Verfügung zu stellen. In der Abbildung 23 ist der schematische Überblick der Konzeption abgebildet und beschreibt den Prozess des *Schema Mappings*. Dieser Prozess ist in drei Schritte gegliedert. Das vorliegende Kapitel, Abschnitt 5.2, orientiert sich am Schema Mapping-Prozess.

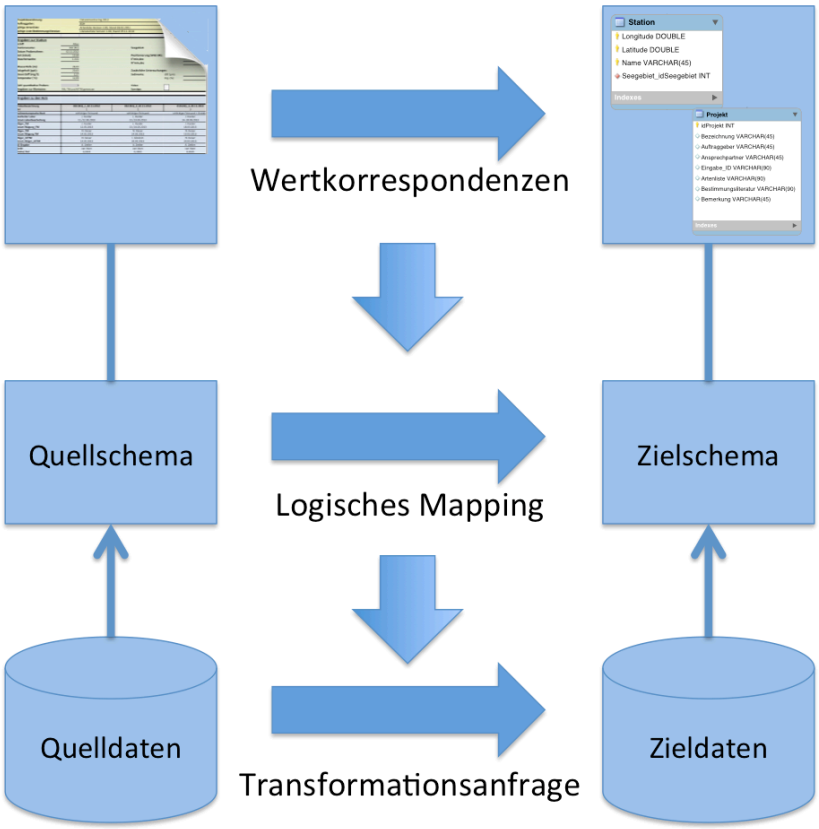


Abbildung 23: Überblick der Konzeption anhand des Schema-Mappings [24]

Die Abbildung 23 zeigt ein Quell- und ein Zielschema. Das Quellschema beschreibt die Daten einer Datenquelle und das Zielschema beschreibt die Daten des Zielsystems. Die Daten sind in der Abbildung 23 als Quelldaten und als Zieldaten gekennzeichnet. Die Aufgabe der vorliegenden Arbeit ist die Transformation der Daten des Quellschemas in die Struktur des Zielschemas. Der erste Schritt des Schema Mappings behandelt das Aufstellen von Wertkorrespondenzen. Wertkorrespondenzen beschreiben eine gerichtete Abbildung von den Attributen des Quellschemas, auf die Attribute des Zielschemas. Die Begriffe *Wertkorrespondenz*

und *Mapping* werden häufig im gleichen Zusammenhang verwendet. Das Mapping beschreibt eine Menge von Wertkorrespondenzen und ist folgendermaßen definiert [24]:

> *„Ein Mapping beschreibt einen semantischen Zusammenhang zwischen Elementen verschiedener Schemata.“*

Im zweiten Schritt werden die zuvor spezifizierten Wertkorrespondenzen als ein logisches Mappings interpretiert bzw. übersetzt. Das logische Mapping ist eine logische Übersetzung der Wertkorrespondenzen. Der dritte Schritt des Schema Mapping-Prozesses beschreibt die Generierung von Transformationsanfragen, mit deren Hilfe die Quelldaten in die Zieldaten überführt werden. Transformationanfragen werden in einer bestimmten Anfragesprache ausgedrückt. Die drei Teilschritte werden in Abschnitt 5.2 anhand der unterschiedlichen Protokolltypen und dem modellierten Zielschema erläutert.
Vorab ist in Abschnitt 5.1 der Entwurf eines einheitlichen Zielschemas notwendig. Das Zielschema hat die Aufgabe, sämtliche Daten in einer einheitlichen Struktur zu speichern. Um das Zielschema modellieren zu können, ist zunächst eine umfangreiche Analyse der zu integrierenden Daten aus den Eingabeprotokollen sowie der Anforderungen an das System, erforderlich. Die Informationen, die das System aufnehmen und verarbeiten soll, werden anhand einer inhaltlichen Analyse in Abschnitt 4.1.1 identifiziert. Anschließend erfolgt in 5.1 die Modellierung des Zielschemas.

5.1. Entwurf des Zielschemas

In diesem Abschnitt wird die Modellierung des Zielschemas beschrieben. Anhand der inhaltlichen Analyse der Datenquellen aus 4.1.1 wird ein geeignetes globales Zielschema entworfen. Die Abbildung 24 verdeutlicht diesen Prozess und zeigt auf der linken Seite die unterschiedlichen Quellschemata. Durch die bereits durchgeführte inhaltliche- und strukturelle Analyse ist festzustellen, dass es sich hierbei um 10 unterschiedliche Excel-Protokolltypen handelt, die jeweils als Quellschema zu betrachten sind. Aus diesen 10 Quellschemata gilt es, ein einheitliches Zielschema, auf der rechten Seite der Abbildung 24, abzuleiten. Das einheitliche Zielschemas wird anhand des ER-Modells beschrieben und liegt im relationalen Datenbankmodell vor.

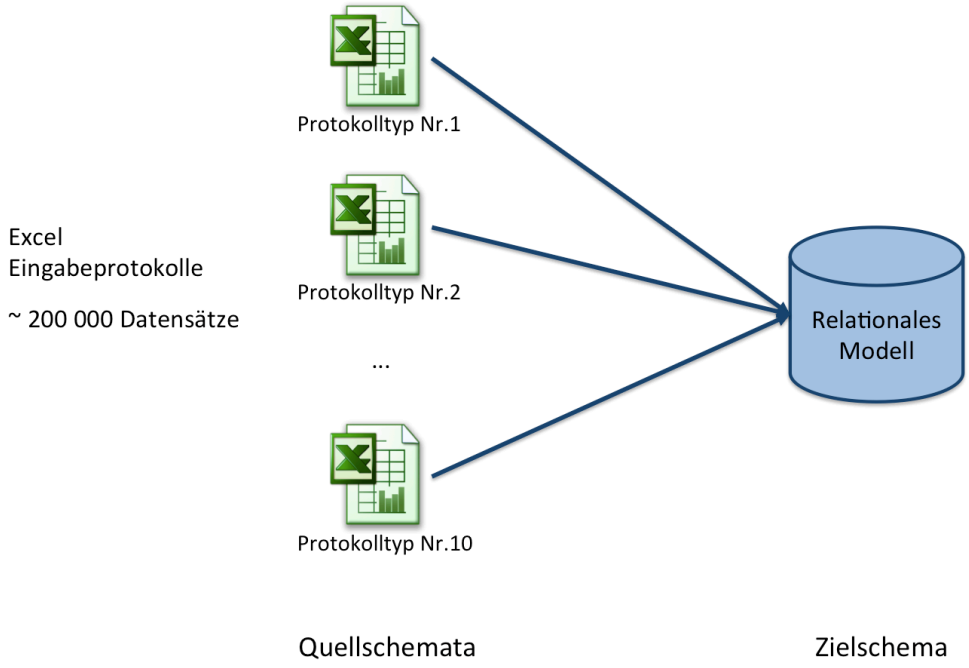


Abbildung 24: Entwurf eines geeigneten Zielschemas

5.1.1. Modellierung

Der erste Schritt in der Modellierung besteht darin, die Anforderungen an das Datenbanksystem zu definieren. Dabei ist es notwendig, die Anwendung zu verstehen, indem sämtliche Objekte der Quelldaten analysiert und identifiziert werden. Unter den Objekten werden Sachverhalte bzw. Gegenstände der realen Welt verstanden und werden anhand der inhaltlichen Analyse der Quelldaten erfasst. Die inhaltliche Analyse ist mit der Anforderungsanalyse gleichzusetzen und wurde bereits in Kapitel 4.1.1 durchgeführt.
Der zweite Schritt in der Datenbankmodellierung besteht darin, auf Basis der inhaltlichen Analyse einen Datenbankentwurf zu erstellen. Dieser Schritt wird als konzeptioneller Datenbankentwurf bezeichnet. Dabei wird das Ziel verfolgt, die aus dem vorherigen Abschnitt gewonnenen Erkenntnisse in einem konkreten Modell darzustellen und genauer zu beschreiben.

In diesem Abschnitt wird die konkrete Modellierung des einheitlichen Zielschemas erläutert. Dabei ist das ER-Modell als Modellierungsverfahren gewählt worden, welches im folgendem spezifiziert wird.
Die wesentlichen Modellierungsstrukturen im ER-Modell sind die *Entities* und die *Relationships* zwischen diesen [21]. Die Entities beschreiben Gegenstände der realen Welt. Die identifizierten Objekte aus 4.1.1 müssen als Entities im ER-Modell dargestellt werden. Weiterhin werden Attribute als Charakteristika, sowie die Beziehungen zwischen den Entities, die ebenfalls durch Attribute beschrieben werden können, im Modell mit angegeben. Ein weiteres Merkmal im ER-Modell sind die Schlüssel, mit deren Hilfe das zugeordnete Entity von allen Entities eindeutig im Modell identifiziert werden kann [21, 29].
Die in den Quellschemata vorhanden Informationen müssen möglichst vollständig in einem einheitlichen Modell dargestellt werden. Da laut der Aufgabenstellung eine MySQL Datenbanklösung gefordert ist, wurde als Tool die MySQL Workbench[31] gewählt, mit deren Hilfe das im folgenden spezifizierte Schema erstellt wurde. Das Tool ist in Abbildung 25 grafisch dargestellt. Die MySQL Workbench bietet einen modellorientierten und visuellen Datenbankentwurf und setzt zudem das relationale Datenbankmodell ein.

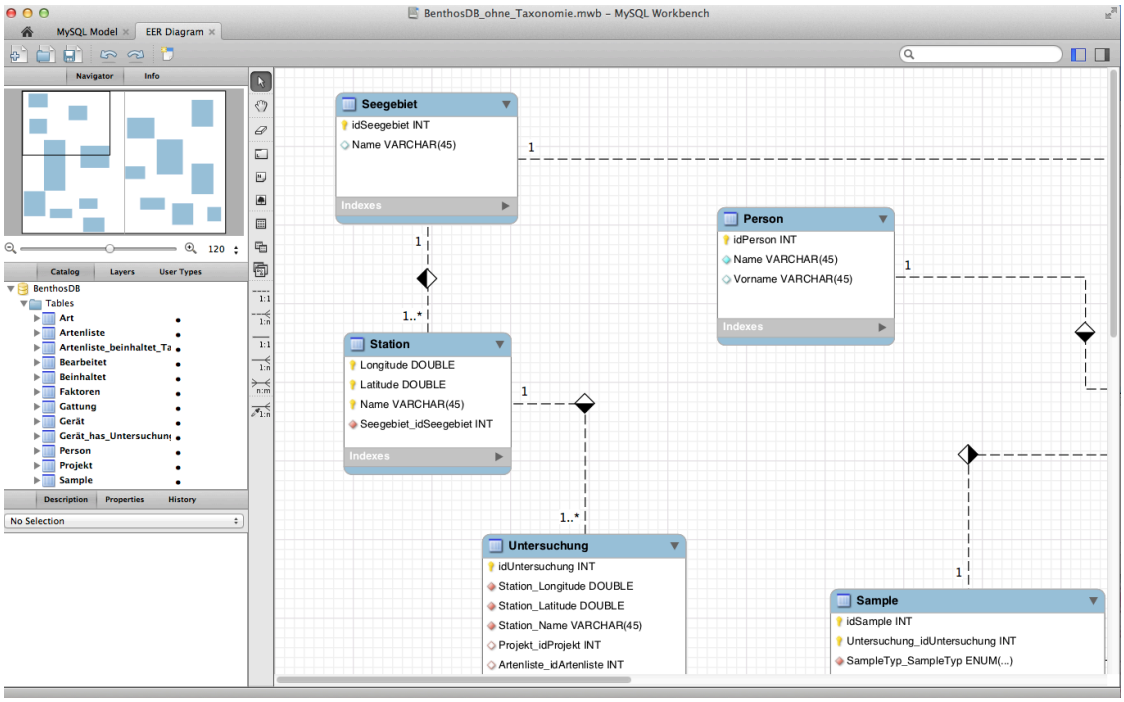


Abbildung 25: Visueller Datenbankentwurf der MySQL Workbench

[31] Die MySQL Workbench ist ein grafisches Entwicklungssystem, mit deren Hilfe das Erstellen, Bearbeiten, Verwalten, Darstellen etc. von Datenbanken möglich ist.

In Tabelle 3 sind sämtliche Entities und deren Primärschlüssel aufgelistet. Die Entscheidung für die Entities ist auf die inhaltliche Analyse der Quellschemata zurückzuführen. Als Schlüssel wurden zumeist künstlich erzeugte Schlüssel gewählt. Dies ist damit zu begründen, dass Schlüssel so kurz wie möglich sein sollten. Dabei eignet sich beispielsweise die Einführung einer "ID". Dieser Wert wird automatisch vom System durch ein Autoinkrementierungsgenerator hochgezählt, sobald ein neuer Datensatz hinzugefügt wird. Dieser Wert wird für jede Entity einmal vergeben. Dadurch kann jedes Tupel in der Datenbank effektiv und eindeutig identifiziert werden. Der Vorteil künstlich erzeugter Schlüssel liegt darin, dass bei der Generierung von Joins[32] nur eine Verbindung über einen generierten Schlüssel notwendig ist. Sind mehrere Primärschlüssel in einer Tabelle angegeben, dann müssen auch über die gesamten Primärschlüssel die Verknüpfungen durchgeführt werden. Dies ist relativ aufwändig und kann durch die Einführung künstlich erzeugter Schlüssel, wie eine "ID", effektiver gelöst werden. Kurze Schlüssel haben außerdem den Vorteil, den Speicherplatz so gering wie möglich zu halten. Zusammengesetzte Schlüssel eignen sich weniger, da diese die Leistung verlangsamen. Außerdem müssen die zusammengesetzten Schlüssel in allen referenzierten Tabellen gleich sein. Dies hat einen höheren Bedarf Speicherplatz zur Folge. In der Praxis kommt es häufig vor, dass sich zusammengesetzte Schlüssel nicht vermeiden lassen. Dies wird im weiteren Verlauf anhand eines Beispiels begründet [9].

Entity	**Primärschlüssel**
Seegebiet	idSeegebiet
Station	Longitude, Latitude, Name
Untersuchung	idUntersuchung
Projekt	idProjekt
Gerät	idGerät
Person	idPerson
Sample	idSample, Untersuchung_idUntersuchung
SampleTyp	SampleTyp
Artenliste	idArtenliste
Taxon	idTaxon
Faktoren	idFaktoren
Gattung	idGattung
Art	idArt

Tabelle 3: Auflistung der beinhalteten Entities mit Primärschlüssel

[32] Über Joins werden Relationen miteinander verbunden.

In der nachfolgenden Tabelle 4 sind alle im ER-Modell auftretenden Beziehungen zwischen den Entities aufgelistet. Dabei treten folgende Beziehungsarten auf, die in *Min-Max-Notation* angegeben sind: ((0,1),(1,1)), ((0,1),(1,N)), ((1,1),(1,N)) und ((1,N),(1,N)). Es existieren eine Reihe an Notationsformen für die Beziehungen im ER-Modell. Die *Min-Max-Notation* bietet den Vorteil, dass an der jeweils beteiligten Entity, durch die Angabe eines Minimal- und Maximalwertes, die Teilnahme der Beziehung in Form einer "Schranke" dargestellt werden kann. Über diese Notation wird angegeben, wie viele Entities mindestens und höchstens an einer Beziehung teilnehmen. Für die ER-Modellierung existiert kein einheitlicher Standard. Die Entscheidung für eine Notationsform liegt beim Modellierer selbst. Die vier auftretenden Beziehungstypen werden im Folgendem erläutert. Das erste Intervall ist der Entity 1 und das zweite Intervall der Entity 2 zugeordnet.

- **((0,1),(1,1))** Die Entity 1 kann keinmal oder einmal an der Beziehung teilnehmen, die Entity 2 nur einmal.
- **((0,1),(1,N))** Die Entity 1 kann keinmal oder einmal an der Beziehung teilnehmen, die Entity 2 ein oder mehrmals.
- **((1,1),(1,N))** Die Entity 1 nimmt einmal an der Beziehung teil, die Entity 2 ein oder mehrmals.
- **((1,N),(1,N))** Beide Entities nehmen jeweils einmal oder mehrmals an der Beziehung teil.

Beziehung	**Beteiligte Entities**
((1,1),(1,N))	Seegebiet, Station
((1,1),(1,N))	Station, Untersuchung
((0,1),(1,N))	Projekt, Untersuchung
((1,N),(1,N))	Gerät, Untersuchung
((1,1),(1,N))	Untersuchung, Sample
((1,N),(1,N))	Person, Sample
((1,N),(1,N))	Sample, Taxon
((1,1),(1,N))	SampleTyp, Sample
((0,1),(1,N))	Artenliste, Untersuchung
((1,N),(1,N))	Artenliste, Taxon
((1,1),(1,N))	Gattung, Art
((0,1),(1,1))	Gattung, Taxon
((0,1),(1,1))	Art, Taxon
((0,1),(1,N))	Taxon, Faktoren
((1,1),(1,N))	Seegebiet, Faktoren

Tabelle 4: Auflistung der Beziehungen zwischen Entities

Die Beziehungen werden im ER-Modell wie folgt umgesetzt:
Bei der ((0,1),(1,1)) Beziehung wird der Primärschlüssel der Entity 1 als Fremdschlüssel in der Entity 2 eingetragen. Selbiges gilt für die Modellierung der ((0,1),(1,N)) und ((1,1),(1,N)) Beziehung. Bei der ((1,N),(1,N)) Beziehung muss automatisch eine neue Tabelle generiert werden, da diese Art der Beziehung nicht direkt umgesetzt werden kann. Die zusätzliche Tabelle enthält die Primärschlüssel der beiden Entities. Diese Schlüssel können als Primärschlüssel oder als Fremdschlüssel in der neuen Entity fungieren. Wird ein künstlicher Schlüssel für die neue Entity erzeugt, dann werden die Schlüssel aus den anderen beiden Entities als Fremdschlüssel in der neuen Entity modelliert. Aus Tabelle 4 geht hervor, dass vier dieser Beziehungstypen modelliert werden müssen. Die neu generierten Entities und deren Schlüssel sind in Tabelle 5 aufgelistet. Durch die zuvor spezifizierten Komponenten, die in Tabelle 3 bis 5 aufgelistet sind, kann ein Zielschema abgeleitet werden. Das vollständige Diagramm ist im Anhang zu finden. Im Folgenden werden zwei Beispiele erläutert.

Beteiligte Entities	**Neu generierte Entity**	**Schlüssel**
Person, Sample	Bearbeitet	idBearbeitet
Sample, Taxon	Beinhaltet	idBeinhaltet
Untersuchung, Gerät	Untersuchung_hat_Gerät	Gerät_idGerät, Untersuchung_idUntersuchung
Artenliste, Taxon	Artenliste_beinhaltete_Taxon	Artenliste_idArtenliste, Taxon_idTaxon

Tabelle 5: Durch ((1,N),(1,N)) Beziehung entstandene Entities

Beispiel 1

Im vorherigem Abschnitt wurden sämtliche Informationen festgehalten, die das Datenbanksystem aufnehmen und verarbeiten soll. Über das ER-Diagramm in der MySQL Workbench, erfolgt eine einheitliche Beschreibung der gewonnenen Anforderungen. In Abbildung 26 ist ein Ausschnitt des modellierten ER-Diagramms dargestellt. Dieser Ausschnitt zeigt die drei Entities *Untersuchung*, *Station* und *Seegebiet*.

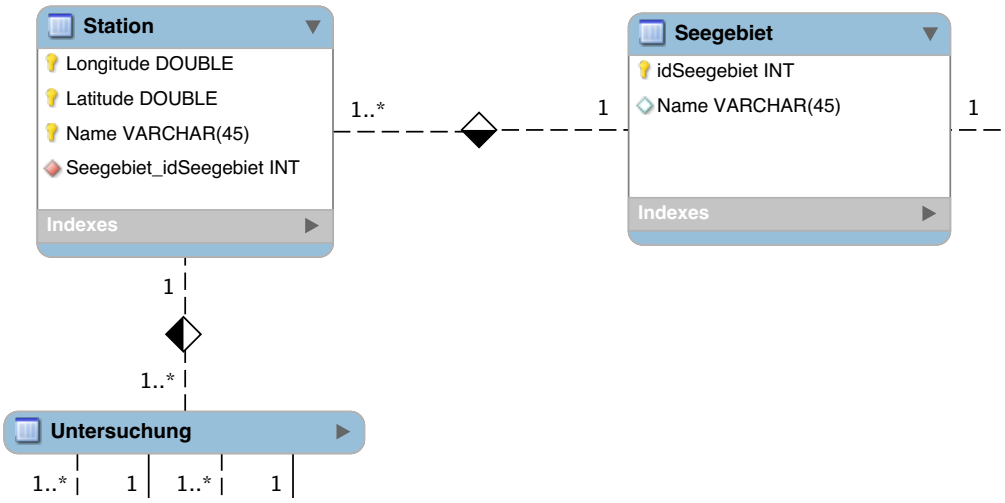


Abbildung 26: Ausschnitt 1 des ER-Modells

Als Objektnotation wurde die von der Workbench angebotene klassische Notation gewählt, die eine übersichtliche Darstellung des Diagramms bietet. Für die Darstellung der Beziehungen zwischen den Entities werden unterschiedliche Notationen angeboten. Da kein einheitlicher Standard für die Darstellung von ER-Diagrammen existiert, kann die Notationsform frei gewählt werden. Hierbei wurde die bereits beschriebene *Min-Max* Notationsform gewählt.

Hauptaufgabe des Datenbanksystems ist die Verwaltung der Eingabeprotokolle pro Messstation. Eine Station hat bestimmte Eigenschaften, Attribute genannt, wie Name und die Positionierung, aufgeteilt in Longitude und Latitude. Aus den Protokollen geht ebenfalls das Seegebiet hervor. Die Entscheidung für die separate Modellierung der Entity *Station* und *Seegebiet* ist auf die eindeutige Speicherung der Daten zurückzuführen. Ein guter Datenbankentwurf besagt, dass keine Redundanzen eines Datensatzes vorhanden sein dürfen. Redundanzen führen dazu, dass ein Datensatz nicht mehr eindeutig in der Datenbank bestimmt werden kann. Einem Seegebiet sind eine Reihe an Stationen zugeordnet. Wird das zugehörige Seegebiet als Attribut in die Entity Station eingetragen, so sind Redundanzen nicht ausgeschlossen. Aus diesem Grund wurde das Seegebiet als eigenständige Entity modelliert. Zwischen den beiden Entities herrscht eine ((1,1),(1,N)) Beziehung.

Weiterhin geht aus der Analyse hervor, dass eine Station eindeutig durch die jeweiligen Koordinaten und dem Namen identifiziert werden kann. Somit ergibt sich ein zusammengesetzter Primärschlüssel. Hierbei wurde kein künstlich erzeugter Schlüssel eingeführt, da das modellierte Datenbankschema für spätere Anwendungen an ein anderes Datenbanksche-

ma angeknüpft werden soll und hierfür die Schlüssel für eine eindeutige Identifikation notwendig sind. Die Entity Seegebiet beinhaltet einen künstlich erzeugter Schlüssel. Aufgrund der vorliegenden ((1,1),(1,N)) Beziehung, wird der Primärschlüssel der Entity Seegebiet als Fremdschlüssel in der Entity Station eingetragen.
In der Abbildung 26 ist eine weitere Beziehung, zwischen der Entity Station und der Entity Untersuchung, dargestellt. Die inhaltliche Analyse der Quelldaten besagt, dass an ein und derselben Station mehrere Untersuchungen stattfinden können. Aus diesem Grund wurde eine ((1,1),(1,N)) Beziehung modelliert. Die drei Primärschlüssel der Entity Station werden in der Entity Untersuchung eingetragen.

Während der Modellierung wurde versucht, den Datenbankentwurf weitestgehend in die Normalformen zu überführen. Dies konnte jedoch in der zur Verfügung stehenden Zeit nicht ausreichend getestet werden. Aus diesem Grund kann nicht gewährleistet werden, dass die Normalformen eingehalten wurden. Durch die Einhaltung der Normalformen können Inkonsistenzen in der Datenbank vermieden werden. Treten Inkonsistenzen in einer Datenbank auf, kann der Datensatz nicht mehr eindeutig bestimmt werden.

Beispiel 2

In Abbildung 27 ist ein weiterer Ausschnitt des modellierten Schemas abgebildet und zeigt die drei Entities *Sample*, *Beinhaltet* und *Taxon*. In diesem Beispiel wird die Modellierung einer ((1,N),(1,N))-Beziehung zwischen zwei Entities beschrieben.

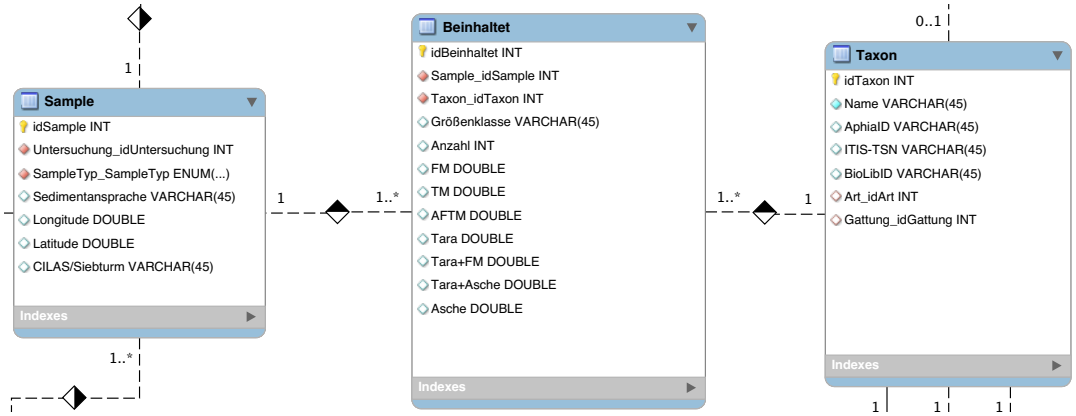


Abbildung 27: Ausschnitt 2 des ER-Modells

Die Entity *Sample* beschreibt die einzelnen genommen Proben an einer Station und bezieht sich auf Hol 1 bis 3, Dredge und Sediment. Jedes Sample wird durch verschiedene Attribute beschrieben. Ein Sample wird durch die Einführung einer ”ID”, als Schlüssel, eindeutig identifiziert. Die ”ID” ist als *auto increment* gekennzeichnet. Eine eindeutige Identifikation des Samples kann nur durch das Modellieren einer ((1,1),(1,N)) Beziehung zwischen Untersuchung und Sample garantiert werden und kann nicht ohne eine Untersuchung existieren. Solche Entities werden *abhängige Entity* genannt. Die Entity Sample enthält noch einen weiteren Fremdschlüssel. Um Redundanzen zu vermeiden, wurde eine neue Entity gebildet, die die Bezeichnungen der Proben (Hol1 bis Hol3, Dredge, Sediment) speichern soll. Der Primärschlüssel dieser Entity wird in Sample als Fremdschlüssel gespeichert. Die Entity Taxon beschreibt die benthischen Organismen. Zwischen den beiden Entities existiert eine ((1,N),(1,N))-Beziehung, da zu jeder Probe eine Reihe an benthischen Organismen protokolliert werden und in diesem Zusammenhang ein bzw. mehrere Samples beteiligt sind. Hierzu muss eine neue Entity mit der Bezeichnung *Beinhaltet* modelliert werden. Diese Entity enthält einen künstlich erzeugten Schlüssel, eine ”ID” und als Fremdschlüssel die Primärschlüssel der beiden beteiligten Entities und beschreibt die gewonnen Informationen der einzelnen Proben zu den protokollierten Taxonen. Dazu zählen Attribute wie Anzahl, Größenklasse, Feuchtgewicht, etc. Jedem Attribut einer Entity ist ein eindeutiger Datentyp zugewiesen.

Nach der Spezifikation des ER-Modell, muss dieses in das relationale Datenmodell umgesetzt werden. Hierbei werden die Objekte in Tabellen und deren Relationen umgesetzt. Dieser Transformationsschritt vom konzeptionellen zum logische Modell erfolgt dabei automatisch durch die MySQL Workbench.

Der letzte Schritt der kompletten Modellierung der Datenbank besteht darin, weitere Anpassungen vorzunehmen. Darunter zählen sogenannte *Unique Keys*. Hierüber können Be-

schränkung, sogenannte *Constraints*, definiert werden, welche keine doppelten Datensätze erlauben. Beispielsweise wurde in der Entity *Seegebiet* das Attribut Name als Unique Key gekennzeichnet. Dies hat zur Folge, dass der Name des Seegebietes nicht doppelt vorkommen kann und somit zwei Datensätze, die den gleichen Namen aufweisen, zwei unterschiedliche IDs (Primärschlüssel) vergeben bekommen. Doppelte Datensätze werden vermieden. Damit kann eine eindeutige Zuordnung und das Auswählen des korrekten Datensatzes gewährleistet werden. In Tabelle 6 sind alle Unique Keys aufgelistet.

Entity	**Als Unique Key gekennzeichnetes Attribut**
Seegebiet	Name
Station	(Longitude, Latitude)
Beinhaltet	(Taxon_idTaxon, Größenklasse)
Gattung	Name
Art	Name
Person	Name

Tabelle 6: Auflistung der Unique Keys

5.1.2. Zusammenfassung

In diesem Kapitel wurde der Entwurf des einheitlichen Zielschemas beschrieben. Dabei wurden sämtliche Entities, Schlüssel, Beziehungen sowie Unique Keys aufgelistet. Während der Modellierung wurden weitestgehend die in 3.2 definierten Schritte eingehalten. Das komplette Schema ist im Anhang zu finden. Anhand zweier Ausschnitte wurde ein Teil des kompletten Schemas erläutert. Das modellierte Datenbankschema ist Voraussetzung für die Integration möglichst aller Quelldaten. Der Prozess der Datenintegration wird im nächsten Abschnitt vorgenommen.

5.2. Datenintegration

Eine Technik zur Überwindung der Heterogenität ist unter anderem das *Schema Mapping*, mit deren Hilfe die Integration möglichst aller relevanten Quelldaten in ein Zielsystem realisiert werden soll. Hierbei werden die heterogenen Excel-Protokolle zu einer einheitlichen Struktur zusammengeführt. Ausgangspunkt der Datenintegration sind die unterschiedlichen Quellschemata und das zuvor modellierte Zielschema.

5.2.1. Festlegung der Wertkorrespondenzen

In der Abbildung 28 ist der erste Prozessschritt des *Schema Mappings* hervorgehoben. Ein wichtiges Konzept des Schema Mappings stellen die *Wertkorrespondenzen* bzw. *Mappings* dar.

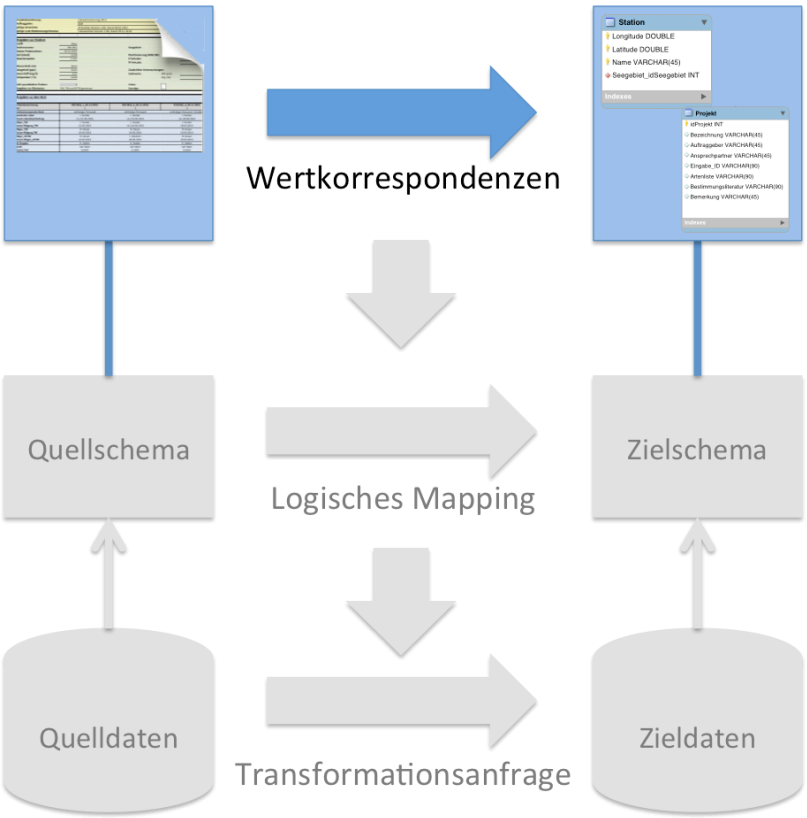


Abbildung 28: Spezifikation der Wertkorrespondenzen zwischen Schemaelementen

Zwischen dem lokalen Schema, dem Quellschema, und dem globalen Schema, Zielschema, werden Abbildungen definiert [32]. Die Begriffe *Wertkorrespondenz* und *Mapping* wurde bereits zu Beginn des Kapitels definiert. Die Wertkorrespondenzen stellen direkte Zuordnungen von Elementen des Quellschemas zu den Elementen eines Zielschemas dar [25].

Die Wertkorrespondenzen können über eine grafische Darstellung repräsentiert werden. Ausgangspunkt sind eine oder mehrere Attribute im Quellschema [24]. Der Ablauf für das Aufstellen von Wertkorrespondenzen ist folgendermaßen darzustellen:

1. Finden der Attribute im Quell- und Zielschema
2. Abbilden (Mappen) der Daten vom Quell- auf Zielschema

Ausgehend von jedem der 10 Excel-Protokolltypen, muss eine gerichtete Abbildung zum Zielschema gefunden werden. Die Wertkorrespondenzen bestimmten explizit, wie die Werte der Attribute des Zielschemas, aus den Werten der Attribute der Quellschemata erzeugt werden. Jedes in der Excel-Datei vorhandene Attribut wird mit einem Attribut des modellierten Zielschemas verknüpft. Anhand dieser Verknüpfung können die Werte zugeordnet werden, sodass möglichst keine Daten verloren gehen.
Es existieren *einfache-* und *mehrwertige Korrespondenzen*. Einfache Korrespondenzen, auch *1:1 Korrespondenz* genannt, verknüpfen genau ein Attribut eines Quellschemas, mit genau einem Attribut eines Zielschemas. Mehrwertige Korrespondenzen hingegen verknüpfen eine Menge von Attributen eines Quellschemas, mit einer Menge von Attributen eines Zielschemas miteinander. Hierbei werden *1:n* sowie *n:1* Korrespondenzen unterschieden [24]. Die *1:n* Korrespondenzen werden konkret an einem Beispiel beschrieben. *n:1* Korrespondenzen treten nicht auf und werden daher vernachlässigt.
Ein Beispiel für eine mehrwertige Korrespondenz, *1:n* Korrespondenz, ist in Abbildung 29 zu finden. Diese Korrespondenz verknüpft ein Attribut des Quellschemas mit zwei Attributen (Name und Vorname) der Zielrelation *Person*. Der zugehörige Wert des Attributes im Quellschema wird verteilt.

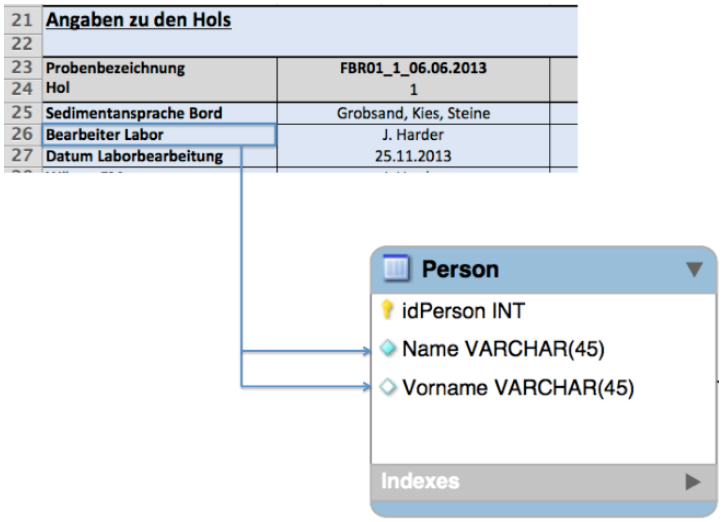


Abbildung 29: Mehrwertige Korrespondenz, Protokolltyp Nr. 10

Im Folgenden werden zwei konkrete Bespiele für die Spezifikation der Wertkorrespondenzen zwischen den Schemaelementen, anhand grafischer Darstellungen, erläutert. Anschließend erfolgt ein Vergleich der Wertkorrespondenzen zweier Protokolltypen.

Beispiel 1

Das Beispiel in Abbildung 30 zeigt eine konkrete Zuordnung von den Excel-Zellen des Zielschemas auf die Attribute einer Relation des Zielschemas.
Die Pfeile bilden eine Wertkorrespondenz ab. Die einzelnen Attribute des Quellschemas (Excel-Protokolltyp Nr. 10) korrespondieren jeweils mit einem Attribut der Zielrelation *Projekt*. Für die Attribute *Bezeichnung* bis einschließlich *Bestimmungsliteratur* der Zielrelation, konnten Wertkorrespondenzen aufgestellt werden. Das Attribut *Bemerkung* der Relation Projekt kann nicht verknüpft werden, da kein zugehöriges Attribut im Quellschema existiert. In dem Beispiel aus Abbildung 30 sind lediglich einfache Korrespondenzen festzustellen.

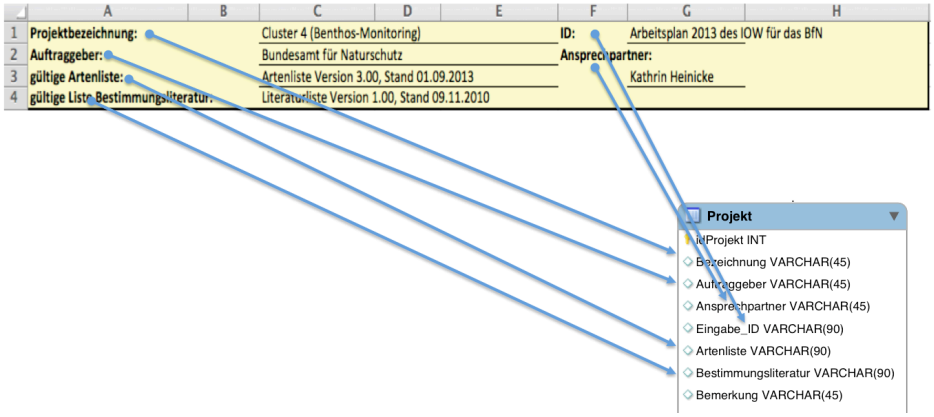


Abbildung 30: Spezifikation der Wertkorrespondenzen zwischen Schemaelementen, Protokolltyp Nr. 10

Beispiel 2

In dem zweiten Beispiel aus Abbildung 31 sind deutlich mehr Zuordnungen zu erkennen. Bei den Informationen des Excel-Protokolltyps Nr. 10 handelt es sich um Stationsdaten, die auf drei Relationen des Zielschemas verteilt sind. Ein Unterschied zum vorherigen Beispiel stellt die modellierte Beziehung zwischen den zwei Relationen, *Station* und *Seegebiet*, des Zielschemas dar. Der Schlüssel aus *Seegebiet* wird in der Relation *Station* als Fremdschlüssel verwendet und muss notwendigerweise von einem Attribut des Excel-Protokolltyps auf das konkrete Attribut *Name* der Zielrelation abgebildet werden.
Weiterhin ist aus der Abbildung 31 zu erkennen, dass sämtliche Zuordnungen der Attribute, aus dem Excel-Protokolltyp, für alle Attribute der Relation *Untersuchung*, die Schlüssel außer Acht gelassen, festgelegt werden konnten.

Vergleich

In der Abbildung 32 ist ein weiteres Beispiel dargestellt, um einen Vergleich zweier Protokolltypen durchführen zu können. Das Beispiel zeigt Wertkorrespondenzen zwischen den Attributen des Protokolltyps Nr. 2 und dem Zielschema. In der Abbildung 32 wird deutlich, dass wesentlich weniger Zuordnungen getroffen werden konnten. Im Gegensatz zum Beispiel

aus Abbildung 31, kann das Attribut *Name* der Relation *Seegebiet* nicht explizit mit einem Attribut des Excel-Protokolls verknüpft werden. An dieser Stelle kann nur durch notwendiges Hintergrundwissen auf diesen Sachverhalt geschlossen werden. Die Verknüpfung ist aus diesem Grund gestrichelt dargestellt. Auch fehlen für weitere Attribute, wie *Uhrzeit*, des Zielschemas konkrete Zuordnungen. Somit können keine Abbildungen generiert werden.

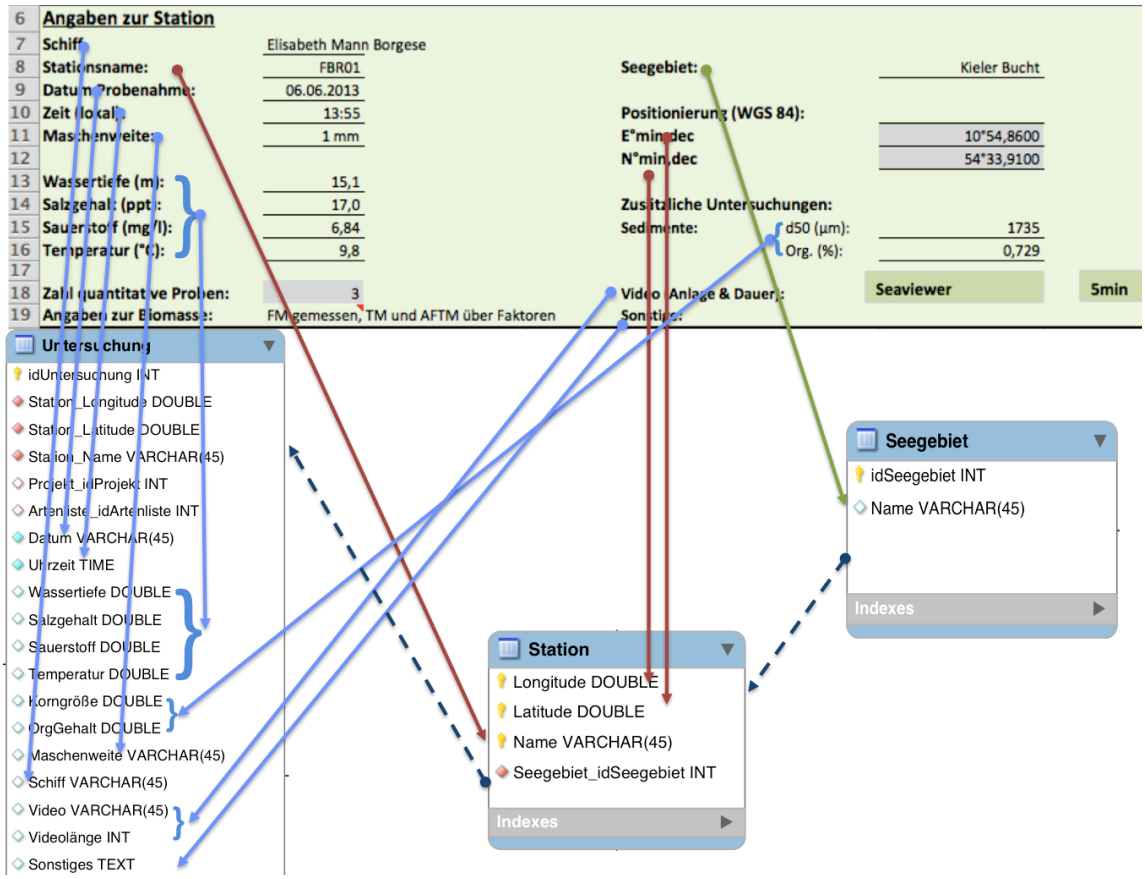


Abbildung 31: Spezifikation der Wertkorrespondenzen zwischen Schemaelementen, Protokolltyp Nr. 10

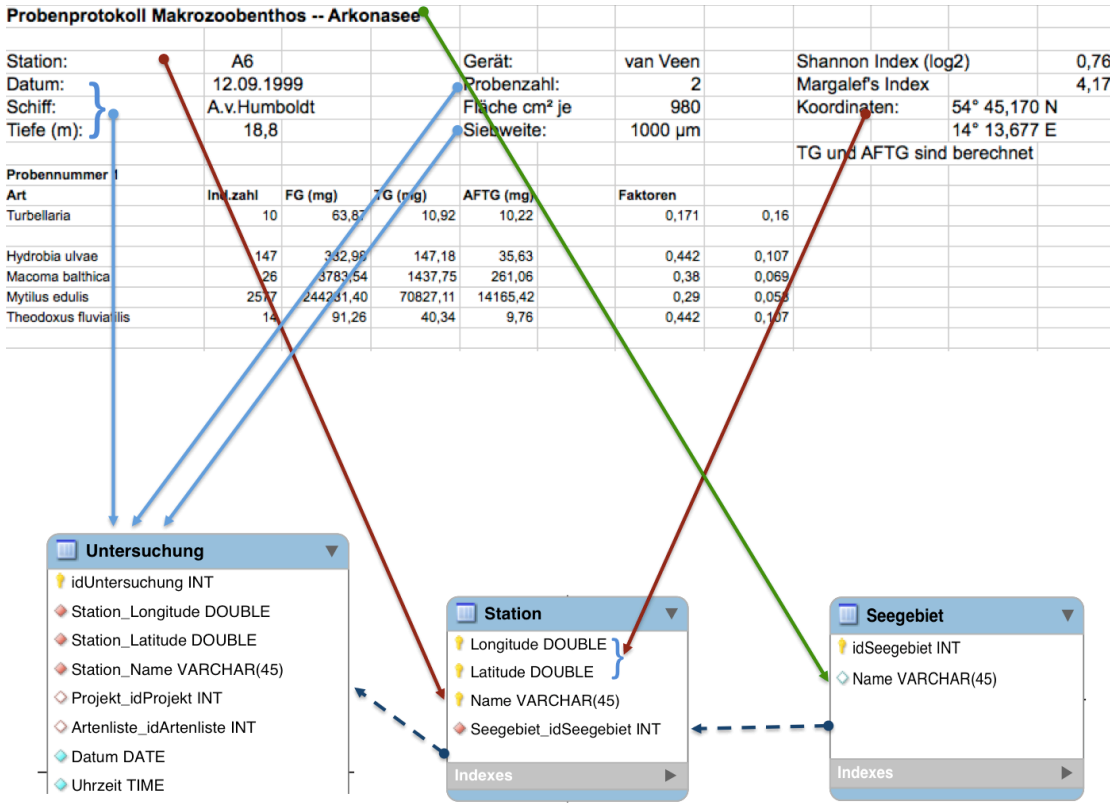


Abbildung 32: Spezifikation der Wertkorrespondenzen zwischen Schemaelementen, Protokolltyp Nr. 2

5.2.2. Adressierung der Bereiche

Der zweite Schritt im Schema Mapping Prozess besteht darin, die zuvor definierten *Wertkorrespondenzen* in ein *logisches Mapping* zu interpretieren. Dieser Prozessschritt ist in der Abbildung 33 hervorgehoben. Der nachfolgende Abschnitt beschreibt drei Ansätze, mit deren Hilfe die exakte Adressierung der zu integrierenden Werte, aus einer Excel-Zelle, möglich ist. Die Adressierung ist notwendig, um die Daten für die Transformation bereitzustellen.

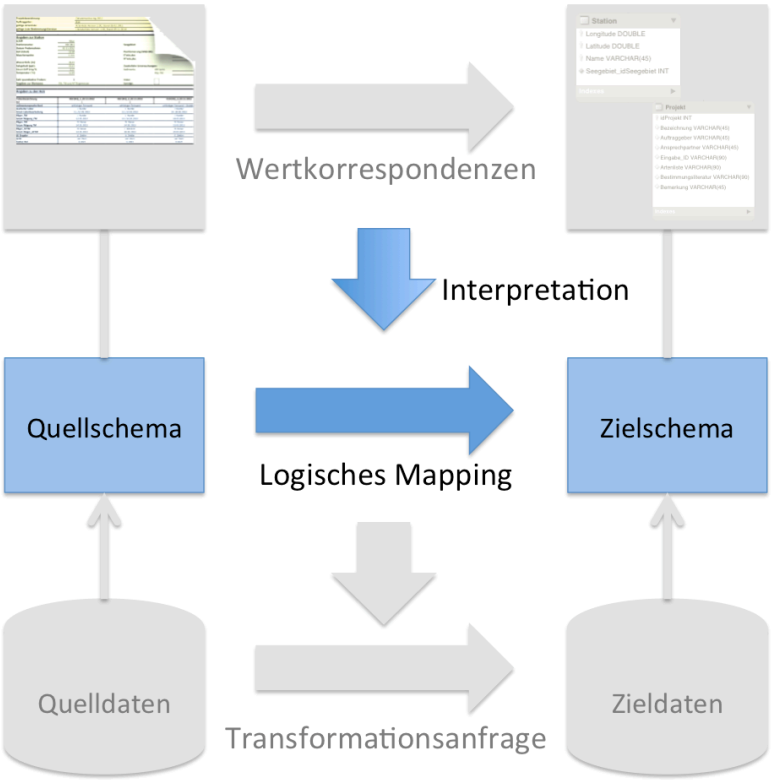


Abbildung 33: Interpretation der Korrespondenzen

1. Ansatz mittels Navigation

Um die Werte aus einer Excel-Zelle adressieren zu können, ist eine Navigation innerhalb einer Excel-Tabelle notwendig. Der erste Ansatz beschreibt diese Navigation, mit deren Hilfe das Lokalisieren von bestimmten Werten möglich ist. Zusätzlich können bestimmte Werte in Abhängigkeit eines anderen Wertes bestimmt werden.
Die Navigation erfolgt über zwei Achsen. Zum einen über die x- und zum anderen über die y-Achse. Die x-Achse repräsentiert in Excel die Spalten-Achse. Hierbei erfolgt eine Navigation in horizontaler Richtung. Die Zeilen-Achse in Excel wird über die y-Achse repräsentiert, über die eine Navigation in vertikaler Richtung erfolgt. Die Navigation in horizontaler Richtung kann nach links (Spalte - a) und nach rechts (Spalte + a), in vertikaler Richtung nach oben (Zeile - b) und nach unten (Zeile + b) erfolgen. Dabei stellen a und b Variablen dar.

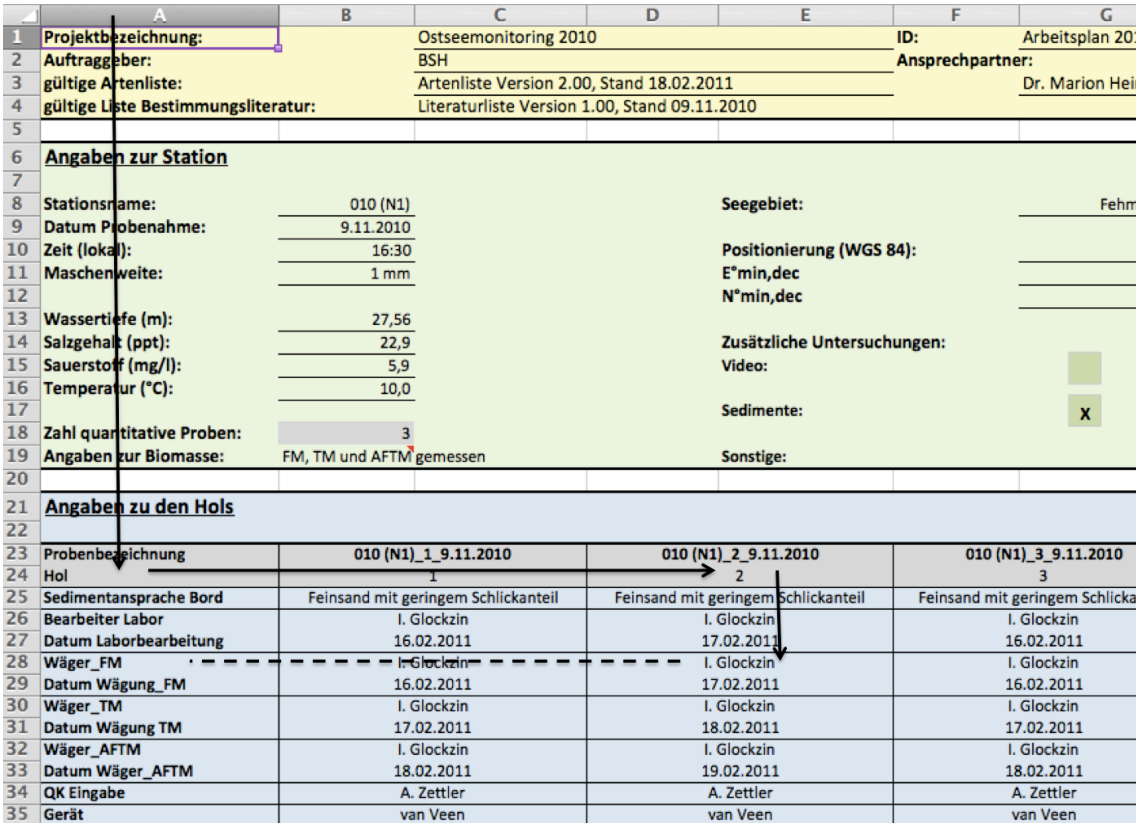
Über die Angabe dieser Variablen kann die Schrittweite in die jeweilige Navigationsrichtung bestimmt werden. Die Navigationsrichtungen sind in der nachfolgenden Tabelle 7 zusammengefasst.

Navigationsrichtung	**Notation in Excel**
x-Achse	Spalte
y-Achse	Zeile
rechts	Spalte + a
links	Spalte - a
unten	Zeile + b
oben	Zeile - b

Tabelle 7: Navigationsrichtungen in Excel

Die Überlegung zum ersten Ansatz ist, ausgehend von der Zelle (Zeile 1 und Spalte 1), die Excel-Tabelle systematisch zu durchlaufen. Aufgrund der Tabellenstruktur kann die Navigation spalten- oder zeilenweise durchgeführt werden. Die Navigation erfolgt über eine Aneinanderreihung von Navigationsschritten. Das Ergebnis ist ein Wert oder eine Menge von Tupel, die möglichst vollständig in die Datenbank zu integrieren sind.
Über die Navigation können relative Pfadabhängigkeiten, durch die Angabe von Bedingungen, realisiert werden. Aufgrund dessen, dass die zu integrierenden Werte nur in Abhängigkeit der jeweiligen Attribute im Quellschema zu lokalisieren sind, muss ein Lokalisierungsschritt Bedingungen enthalten. Diese Bedingungen beinhalten den Namen des abhängigen Attributes im Quellschema.
Um diese theoretischen Überlegungen realisieren zu können, muss eine allgemeine Lösung gefunden werden. Für diese allgemeine Lösung ist eine formale Beschreibungssprache notwendig, die die Adressierung aller relevanten Daten aus einer bzw. mehrerer Zellen ermöglicht.

Die Abbildung 34 illustriert eine Navigation innerhalb einer Excel-Tabelle und verdeutlicht eine relative Pfadabhängigkeit, da der zu lokalisierende Wert von drei Attributen, speziell von drei in einer Excel-Zelle befindlichen Werten, abhängig ist. Gesucht wird der Name des *Wäger FM* für *Hol 2*.



	A	B	C	D	E	F	G
1	Projektbezeichnung:		Ostseemonitoring 2010			ID:	Arbeitsplan 20
2	Auftraggeber:		BSH			Ansprechpartner:	
3	gültige Artenliste:		Artenliste Version 2.00, Stand 18.02.2011				Dr. Marion Hei
4	gültige Liste Bestimmungsliteratur:		Literaturliste Version 1.00, Stand 09.11.2010				
5							
6	**Angaben zur Station**						
7							
8	Stationsname:	010 (N1)			Seegebiet:		Fehm
9	Datum Probenahme:	9.11.2010					
10	Zeit (lokal):	16:30			Positionierung (WGS 84):		
11	Maschenweite:	1 mm			E°min,dec		
12					N°min,dec		
13	Wassertiefe (m):	27,56					
14	Salzgehalt (ppt):	22,9			Zusätzliche Untersuchungen:		
15	Sauerstoff (mg/l):	5,9			Video:		
16	Temperatur (°C):	10,0					
17					Sedimente:		X
18	Zahl quantitative Proben:	3					
19	Angaben zur Biomasse:	FM, TM und AFTM gemessen			Sonstige:		
20							
21	**Angaben zu den Hols**						
22							
23	Probenbezeichnung	010 (N1)_1_9.11.2010		010 (N1)_2_9.11.2010		010 (N1)_3_9.11.2010	
24	Hol	1		2		3	
25	Sedimentansprache Bord	Feinsand mit geringem Schlickanteil		Feinsand mit geringem Schlickanteil		Feinsand mit geringem Schlicka	
26	Bearbeiter Labor	I. Glockzin		I. Glockzin		I. Glockzin	
27	Datum Laborbearbeitung	16.02.2011		17.02.2011		16.02.2011	
28	Wäger_FM	I. Glockzin		I. Glockzin		I. Glockzin	
29	Datum Wägung_FM	16.02.2011		17.02.2011		16.02.2011	
30	Wäger_TM	I. Glockzin		I. Glockzin		I. Glockzin	
31	Datum Wägung TM	17.02.2011		18.02.2011		17.02.2011	
32	Wäger_AFTM	I. Glockzin		I. Glockzin		I. Glockzin	
33	Datum Wäger_AFTM	18.02.2011		19.02.2011		18.02.2011	
34	QK Eingabe	A. Zettler		A. Zettler		A. Zettler	
35	Gerät	van Veen		van Veen		van Veen	

Abbildung 34: Navigation durch das Excel-Dokument

Der Ablauf für die Lokalisierung des Wertes ist folgendermaßen definiert:

1. Der Startpunkt der Navigation ist die erste Zelle (Zeile 1 und Spalte 1).
2. Die Navigation erfolgt zunächst entlang der **y-Achse**, bis der Wert *Hol* lokalisiert wurde.
3. Ausgehend von der erreichten Position aus Schritt 2, **Spalte + a** Schritte weiter navigieren, bis der Wert *2* lokalisiert wurde.
4. Ausgehend von der erreichten Position aus Schritt 3, **Zeile + b** Schritte weiter navigieren.
5. Rücknavigation: **Spalte - a** Schritte zurück navigieren, bis der Wert *Wäger_FM* lokalisiert wurde. In dieser Zeile befindet sich, ausgehend von der Spalte aus Schritt 3, der zu integrierende Wert.

Durch vordefinierte Muster werden die Navigationsrichtungen der jeweils zu lokalisierenden Werte, aus Schritte 2 bis 5, einer Excel-Zelle vorgegeben. Die Muster, engl. Patterns, bilden sogenannte Lösungsschablonen ab, die die Navigationsrichtung für die Lokalisierung von Werten, in Abhängigkeit anderer Werte, beschreiben. Die vier möglichen Navigationsrichtungen, die zuvor in Tabelle 7 aufgelistet wurden, sind Hauptbestandteil der aufgestellten Muster. Um den Lösungsansatz für die Adressierung der zu integrierenden Werte über eine Navigation zu vervollständigen, ist das explizite Definieren solcher Muster für jeden der 10 unterschiedlichen Protokolltypen unumgänglich. Die Abbildung 35 zeigt ein vordefiniertes Muster. Mit diesem Muster können sämtliche Werte, die für die Attribute der Relation *Projekt* benötigt werden, in der Excel-Tabelle adressiert werden. Das Muster bezieht sich dabei auf den Protokolltyp Nr. 10. Das komplette Muster für den Protokolltyp Nr. 10 ist im Anhang aufgelistet.

Attributbezeichnung Zielschema	Attributbezeichnung Quellschema	Navigation zum Wert
Bezeichnung	Projektbezeichnung:	Spalte + a
Auftraggeber	Auftraggeber:	Spalte + a
Ansprechpartner	Ansprechpartner:	1. Spalte + a 2. Zeile + b
Eingabe_ID	ID:	Spalte + a
Artenliste	gültige Artenliste:	Spalte + a
Bestimmungsliteratur	gültige Liste Bestimmungsliteratur:	Spalte + a
Bemerkung	-	-

Abbildung 35: Muster, um die zu integrierenden Werte für die Relation ”Projekt” des Zielschemas zu adressieren; Muster bezieht sich auf den Excel-Protokolltyp Nr. 10

Die Tabelle in Abbildung 35 beinhaltet drei Spalten. Die erste Spalte listet die Namen der Attribute des Zielschemas, die zweite Spalte die Namen der Attribute des Quellschemas, speziell den Excel-Protokollen, auf. Für die Adressierung der Werte muss die konkrete Bezeichnung des Attributes im Excel-Dokument angegeben werden, da sonst keine exakte Lokalisierung der abhängigen Werte gewährleistet werden kann. Das Suchen nach konkreten Namen in Excel kann nur dann korrekt erfolgen, wenn der Name im Excel-Dokument vorhanden ist. Dabei müssen auch Sonderzeichen beachtet werden. In der dritten Spalte sind die jeweiligen Navigationsrichtungen aufgelistet, um den zu integrierenden Wert, in Abhängigkeit des angegebenen Attributes des Quellschemas, zu lokalisieren. Ein Wert kann auch über mehrere Navigationsschritte erreicht werden. Diese Navigationsschritte müssen zusammen angegeben werden, wie in Zeile 4, Spalte 3 der Tabelle aus Abbildung 35 ersichtlich ist. Der Wert kann nur über zwei Navigationsschritte erreicht werden.

Beispiel aus Abbildung 34

Die aus der Abbildung 34 dargestellte Navigation wird mit den aus der Abbildung 36 aufgelisteten Lokalisierungsschritten realisiert. Der Ablauf der Lokalisierung wurde bereits in fünf Schritten zu Beginn des Abschnittes beschrieben. Der Ablauf zur Lokalisierung wird in der Abbildung 36 zusammengefasst.

Schritt Nr.	Iteration Achse	Navigations-richtung	Navigation bis...	Ergebnis Excel-Zelle
1	Y-Achse	Zeile + b	„Hol“	C[i,j]
2	X-Achse	Spalte + a	„2“	C[i,j + 2]
3	Y-Achse	Zeile + b, bis (Spalte – a) = „Wäger_FM“	WERT	C[i + 4,j + 2]= WERT

Abbildung 36: Aufgelistete Navigationsschritte, um den Wert für ”Wäger_FM” bezüglich ”Hol 1” zu lokalisieren; Protokolltyp Nr. 10

Die zweite Spalte gibt die Achse an, über die jeweils iteriert werden muss. Der erste Schritt beschreibt die Iteration über die Y-Achse. Um den Wert ”Hol” zu lokalisieren, müssen zunächst sämtliche Spalten (Iteration entlang der y-Achse) durchlaufen werden. Die dritte Spalte gibt die allgemeine Navigationsrichtung an, die bereits zu Beginn des Kapitels definiert wurde. Die vierte Spalte listet das Attribut des Quellschemas auf, welches in dem jeweiligen Navigationsschritt zu lokalisieren ist. Die erreichte Position der Excel-Zelle, in der der zu suchende Wert gespeichert ist, wird in der letzten Spalte angegeben.
Um den Wert ”2” zu lokalisieren, muss über die Zeile (x-Achse) iteriert werden. Über den dritten Navigationsschritt, der Iteration über die y-Achse wird der zu integrierende Wert adressiert. Bei diesem Schritt muss darauf geachtet werde, dass über die bedingte Rücknavigation (Spalte - a) der Wert ”Wäger_FM” lokalisiert werden muss. Denn nur in Abhängigkeit dieses Wertes kann der zu integrierende Wert adressiert werden. Die erreichte Zielposition der Excel-Zelle ist **C[i + 4,j + 2]**.

Weiteres Beispiel

Die nachfolgende Tabelle in Abbildung 37 zeigt eine weitere Navigation, in welcher der Wert eines Attributwertpaares in drei Schritten adressiert wird. Die Navigation wird in der nachfolgenden Abbildung 38 grafische veranschaulicht. Die Navigation wird in Abbildung 38 grafisch veranschaulicht.

Schritt Nr.	Iteration Achse	Navigations-richtung	Navigation bis...	Ergebnis Excel-Zelle
1	Y-Achse	Zeile + b	„Ansprechpartner“	C[i,j]
2	X-Achse	Spalte + a	-	C[i,j + 1]
3	Y-Achse	Zeile + b	WERT	C[i + 1,j + 1]= WERT

Abbildung 37: Aufgelistete Navigationsschritte, um den Wert für ”Ansprechpartner” zu lokalisieren; Protokolltyp Nr. 10

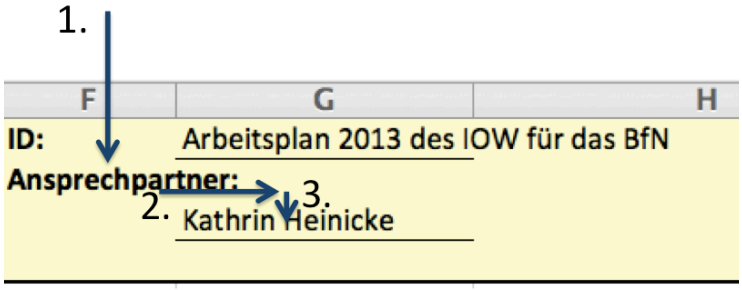


Abbildung 38: Grafische Darstellung der Navigation; Protokolltyp Nr. 10

Gesucht wird der Wert des Attributes ”Ansprechpartner”. Die drei Schritte sind in der Abbildung 38 dargestellt. Nachdem der Wert ”Ansprechpartner”, mittels der Iteration über die y-Achse (Schritt 1) ermittelt wurde, ist der zugehörige Wert über zwei folgende Navigationsschritte (Schritt 2 und 3) zu erreichen.

Zusammenfassung

Die Lokalisierungsschritte sind durch die zu definierenden Muster für jeden Protokolltyp festgelegt, um alle Elemente im Excel-Dokument zu lokalisieren. Die notwendigen Bezeichnungen der Attribute sind bereits durch das Datenbankschema, vorgegeben. Basierend auf den Attributwertpaaren, können Muster spezifiziert werden. Die Tabelle 7 zeigte bereits die grundlegenden Navigationsschritte.

Der zuvor spezifizierte Ansatz beschreibt eine theoretische Lösung, um Werte in einem Excel-Dokument, über eine relative Adressierung, zu lokalisieren. Diese Adressierung wird

mittels einer Navigation innerhalb einer Excel-Tabelle durchgeführt. Dabei wurden die allgemeinen Navigationsrichtungen, die zu spezifizierenden Muster sowie zwei Beispiele erläutert. Um die Adressierung so effektiv wie möglich zu realisieren, müssen im Vorfeld Muster für die verschiedenen Protokolltypen definiert werden. Anhand dieser Muster können die Werte lokalisiert werden. Die Muster geben lediglich die Navigationsrichtung, nicht aber die Schrittweite an. Da nach den jeweiligen Attributbezeichnungen gesucht wird, kann die Schrittweite vernachlässigt werden. Dies hat den Vorteil, dass bei auftretenden Fehlern im Excel-Dokument, falls ein Wert an einer fehlerhaften Position gespeichert ist, trotz alledem der zu integrierende Wert lokalisiert und adressiert werden kann. Der einzige Nachteil liegt darin, dass der zu lokalisierende Wert nur adressiert werden kann, falls dieser sich noch in der gleichen Navigationsrichtung befindet. Das heißt, dass sich die Schrittweite beliebig verändern kann, nicht aber die allgemeine Richtung, z.B von x- auf y-Achse oder umgekehrt, dann kann keine korrekte Adressierung sichergestellt werden.
Je mehr Daten in einem Protokoll gespeichert sind, desto größer ist der Aufwand der Navigation sowie die Spezifikation der Muster. Dabei ist eine exakte Beschreibung notwendig, welche Navigation in welchem Schritt erforderlich ist, um gewährleisten zu können, dass alle Werte korrekt adressiert werden. Kann dies nicht gewährleistet werden, so können falsche Werte in das Zielsystem integrierte werden.
Um den Ansatz umsetzen und testen zu können, müssen die theoretischen Überlegungen, bezüglich der Navigation, in einer konkreten Programmiersprache umgesetzt werden. Das Aufstellen einer neuen Beschreibungssprache stellt eine Herausforderung dar und ist in der zur Verfügung stehenden Zeit nicht zu lösen. Aus diesem Grund können schon vorhandene Beschreibungssprachen herangezogen werden. Der beschriebene Ansatz der Navigation erinnert an die Sprache XPath. Diese Abfragesprache unterstützt das Formulieren von Bedingungen, um Abhängigkeiten flexibler realisieren zu können und in einem Ausdruck mehrere Prozessschritte zu vereinen und über die Ausdrücke die zu dem Attributwertpaar gehörenden Werte zu lokalisieren. In Abschnitt 5.2.2 wird ein weiterer Ansatz, mittels der Sprache XPath, für die Adressierung der Werte vorgestellt.
Weiterhin ist zu diskutieren, warum die zu integrierenden Werte nicht über die Sprache SQL ausgelesen werden können. Da die Werte in Excel in einer tabellarischen Struktur angeordnet sind, könnten die Werte theoretisch über SQL adressiert werden. SQL ermöglicht die direkte Adressierung von Zeilen und Spalten. Da kein Datenbankschema vorhanden ist und die Werte in den vorliegenden Excel-Protokollen nicht in einer einheitlichen Struktur gespeichert sind, sondern die Positionen der Attributwertpaare variieren, ist es nicht möglich, diese Werte über SQL zu adressieren. Die fehlende einheitliche Struktur, sowie das Fehlen eines Datenbankschemas führt dazu, dass der Anwender vor der Adressierung erkennen muss, an welcher Position der spezifische Wert zu lokalisieren ist. Anders bei der eigentlichen Navigation, hierbei wird ausgehend vom Attribut eine Navigation durchgeführt. Bei SQL müssten direkt die Positionen, also absolute Adressierungen angegeben werden. Mittels SQL ist keine Adressierung möglich.
Für die Adressierung der benötigten Werte können bereits vorhandene Ansätze genutzt werden. Einer dieser Ansätze wird im folgenden Abschnitt beschrieben.

2. Ansatz über Doush & Wang

Dieser Ansatz beschreibt die Adressierung von Excel-Zellen, anhand der Forschungsarbeiten von Doush und Wang. Die Grundlagen wurden bereits in Kapitel 3.3 erläutert. Der Ansatz verwendet speziell den von Doush definierten Algorithmus aus [11] sowie die Klassifikation von Excel-Zellen.
Der Algorithmus nutzt das Konzept der *ähnlichen Formatierung* und das *Erkennen von Grenzen*, um die funktionalen Komponenten einer Tabelle voneinander abgrenzen zu können. Weiterhin sind für den Ansatz die Merkmale der Zellen in einer Tabelle sowie der Zusammenhang mit den umliegenden, angrenzenden Zellen erforderlich.
Die Aufgabe dieses Ansatzes ist die Werte aus den Excel-Zellen auszulesen. Der Ansatz beschreibt, wie semantisch zusammengehörige Bereiche bestimmt werden können. Ein Bereich kann ein Wert oder eine Menge von Werten sein. Um zwischen den Bereichen differenzieren zu können, wird zusätzlich die von Wang aus [36] definierte Tabellenstruktur verwendet.
Der Ansatz nutzt zusätzlich die Notation einer Excel-Zelle, die bereits in 3.3.1 definiert wurde. Die formale Adressierung einer Zelle ist in (6) aufgestellt und wird benötigt, um eine Zelle in einer Excel-Tabelle zu lokalisieren.

Zelle: Blatt, Zeile, Spalte (6)

Über die Notation aus (6), mit Angabe des Tabellenblatts, engl. worksheet sowie Zeile und Spalte , kann eine Zelle eindeutig bestimmt werden. Über die folgende Notation in (7) kann die Adresse einer Zelle identifiziert werden, um auf einen bestimmten Wert zu referenzieren.

sheetname.C[i,j] (7)

Die zu adressierenden Daten sind in den **Zellen** gespeichert. Jede Zelle **C[i,j]** hat eine bestimmte Position in einem Tabellenblatt. Eine Zelle hat sowohl einen semantischen, als auch einen physischen Typ. Der semantische Typ beschreibt Informationen wie z.B. eine Tabellenüberschrift, einen Namen oder eine Formel. Der physische Typ beschreibt konkret den Datentyp wie Text, Datum, Zahl etc. Die Identifikation des jeweiligen Typs ist notwendig. Über diese Information können die zu integrierenden Daten ermittelt werden. Die Eigenschaften einer Zelle und deren Zusammenhang mit den umliegenden Zellen ist hilfreich, um verschiedene funktionale Komponenten einer Tabelle zu identifizieren. Bereiche können dabei über die von Excel zur Verfügung stehende Methode *Range* angesprochen werden. Über die Methode kann zum einen eine Zelle *Range(A1)* und zum anderen ein gesamter Bereich an Zellen *Range (A1:B10)* ausgewählt werden. Um eine Spalte auszuwählen wird *Range(A:A)* und um eine Zeile auszuwählen *Range(2:2)* genutzt. Auch lassen sich gesamte Spalten oder Zeilen angeben.

Beispiel:

Adressierung der Spalte B, C, E und F: *Range(B:B, C:C, E:E, F:F)*
Adressierung der Zeile 1 bis 8: *Range(1:8)*

Soll der Bereich, der zuvor über die Methode *Range* adressiert wurde, ausgewählt werden, dann muss der Befehl *select* verwendet werden.

```
1        Selektieren eines Wertes:
2        select  sheetname.C[i,j].value
3
4        Selektieren eines Bereiches:
5        select  sheetname.Range(C(i,j):C(k,l)).value
```

Listing 1: Adressierung der Werte aus Excel-Zelle

Die zuvor angegebene Adressierung ist statisch. Das heißt, dass die Position, der Zeilen- und Spaltenindex, der Werte bekannt sein muss. Dieser Lösungsansatz ist für die vorliegende Arbeit nicht geeignet, da die Positionen der Werte für einen zu definierenden Bereich nicht im Vorfeld klar sind und sich außerdem in Abhängigkeit anderer Zellen ergeben. Für die Lösung wird eine relative Adressierung notwendig. Für die vorliegende Aufgabe werden allerdings Abläufe benötigt, mit deren Hilfe die Adressierung realisiert werden kann.. Die Position der Zelle des zu integrierenden Wertes ergibt sich aus der relativen Abhängigkeit eines anderen Wertes in einer Zelle.

Die Abbildung 39 illustriert, dass die Zelle mit dem zu integrierendem Wert nur in Abhängigkeit der Zelle mit dem Wert "Temperatur:" zu lokalisieren ist. Aus diesem Grund muss vorab geklärt werden, an welcher Position im Excel-Dokument der Wert steht. Von dieser Position aus kann der dazugehörige Wert ermittelt werden.

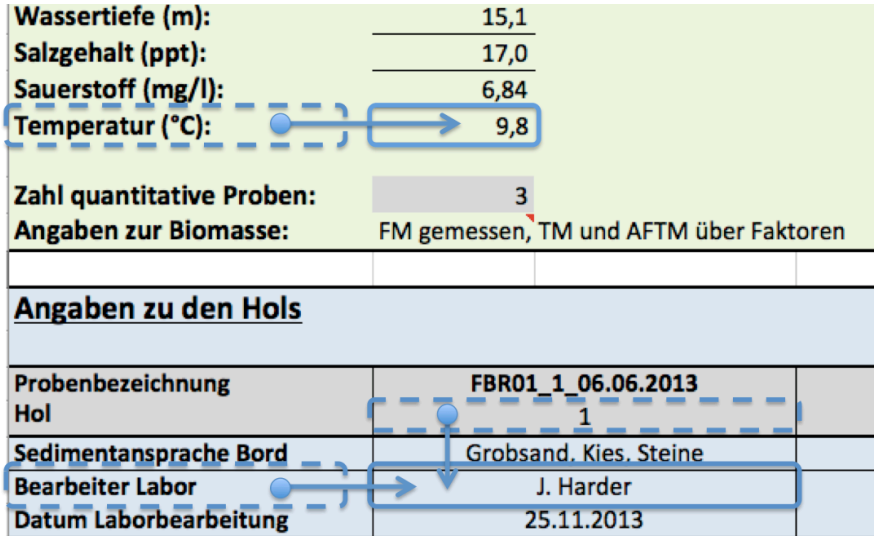


Abbildung 39: Abhängigkeit der zu adressierenden Werte

Für diesen Ablauf ist das Einführen von Variablen zwingend notwendig. Über die *Range* Methode ist es möglich, Variablen zu definieren und zu verwenden. Mittels Schleifen lassen sich bestimmte Zeilen und Spalten nacheinander abarbeiten. Dieser Vorgang ist in zwei Schritten zu lösen:

1. Festlegen der Variablen
2. Schleifen definieren

Zusätzlich wird eine Kombination von der Methode **Range** und **Cell** zu **Range(C[i,j])** benötigt, um den zu integrierenden Wert einer Zelle mit dem beinhalteten Wert zu adressieren. Der Algorithmus von Doush in 3.3.1 verwendet die Klassifikation von Zellen (**Header**-, **Title**- und **Data Cell**) sowie die spezifizierten Merkmale einer Zelle.

Mit der Einführung zweier Variablen x und y können Teile von bzw. mehrere Zeilen und Spalten adressiert werden. Für das Finden eines Wertes sind die benachbarten Zellen zwingend notwendig. Diese benachbarten Zellen können über die definierten Variablen angesteuert werden. Der Ausdruck für das ansteuern von benachbarten Zellen, in Richtung der Zeile (über die Variable x) oder Spalte (über die Variable y), wird in (8) und (9) verdeutlicht.

$$\mathbf{C[i,j];\ x >= 0;\ y >= 0;} \tag{8}$$

$$\mathbf{C[i\ +/-\ x\ ,\ j\ +/-\ y];} \tag{9}$$

Dieser Ausdruck ist in Abbildung 40 grafisch dargestellt, dabei ist x = 0 und y = 1. Die Zelle C[i,j] beinhaltet den String "Temperatur". Die angesteuerte Zelle muss eine Header Cell sein. Zu dieser Header Cell wird die dazugehörige Data Cell gesucht. Die Data Cell beinhaltet den zu integrierenden Wert. Dabei müssen sämtliche benachbarte Zellen angesteuert und bezüglich ihrer Klassifikation untersucht werden. Der Wert der Temperatur ist ausgehend von der Zelle C[i,j], in der darauffolgenden Zelle C[i,j + 1] gespeichert. Der Spaltenindex wird um eins inkrementiert.

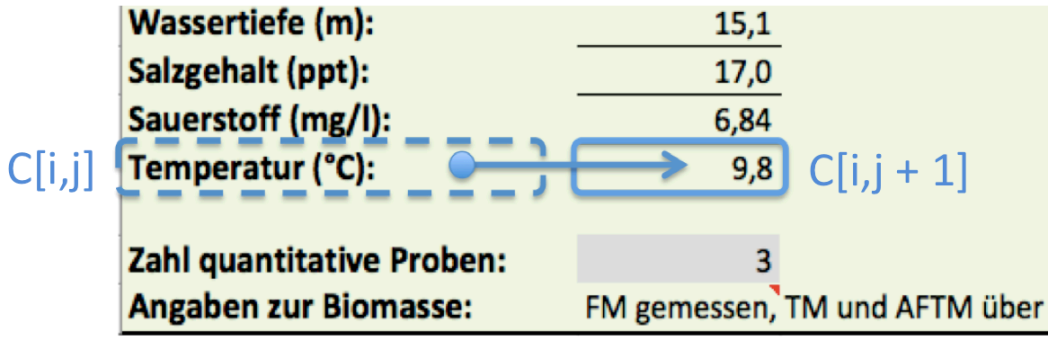


Abbildung 40: Belegung der Zelle

Der spezifizierte Algorithmus stellt lediglich einen theoretischen Lösungsansatz dar und wurde, bezüglich der Funktionalität, nicht in einer konkreten Programmiersprache getestet.

Zusammenfassung

Die zuvor beschriebene Lösung eignet sich nicht für die Adressierung von Werten aus einer Excel-Zelle, um diese anschließend für die Integration zur Verfügung zu stellen. Mit diesem Ansatz ist es nicht möglich, Werte aus relativen Abhängigkeiten zu lokalisieren sowie konkrete Bedingungen hierfür festzulegen. Es lassen sich lediglich Werte bestimmen, die von einem Wert, dem zugehörigen Attribut, abhängig sind. Werden komplexe Pfade benötigt, dann ist der Ansatz nicht in der Lage, dieses Problem zu lösen und daher für die Datenintegration der Arbeit ungeeignet.

3. Ansatz über XPath

Der dritte Ansatz beschäftigt sich mit einer Adressierung über XPath-Ausdrücke. XPath ist eine Abfragesprache für XML-Dokumente, mit deren Hilfe die Adressierung von Teilen eines XML-Dokumentes möglich ist.
Die Idee liegt darin, die zu integrierenden Werte abzufragen und gleichzeitig über XPath-Ausdrücke zu adressieren. Hierzu werden vorab sämtliche Excel-Eingabeprotokolle in ein per XPath adressierbares Dateiformat (XML-Dateiformat) transformiert. Diese Transformation stellt keine Schwierigkeit dar, da Excel die Funktion beinhaltet, Dokumente in das XML-Dateiformat abzulegen. Über XML werden semistrukturierte Daten[33] in einer Hierarchie dargestellt.
Ein XML-Dokument kann als Baum betrachtet werden. Die Baumstruktur bildet die Grundlage von XPath und stellt dabei die Informationseinheiten dar, die in einem XML-Dokument dargestellt sind und definiert hierbei eine Sicht auf das XML-Dokument. Die Abfragesprache XPath differenziert zwischen unterschiedlichen Knotentypen. Knotentypen sind unter anderem der Wurzel-, Element-, Attribut-, Text- sowie Namensraumknoten. Der Wurzelknoten stellt den ersten Knoten der Baumstruktur dar. Ausgehend von diesem Knoten zweigen sich weitere Knoten ab [20]. Die wichtigste Struktureinheit bildet das *Element*, in welchem sich die Daten befinden. Elemente können weitere Elemente enthalten, dadurch wird eine Hierarchie gebildet.
In Abbildung 41 ist ein Ausschnitt der Baumstruktur des transformierten Excel-Protokolltyps Nr. 10, in das XML-Dateiformat, abgebildet. Die abgebildete Baumstruktur beinhaltet eine Reihe an Knoten. Der Wurzelknoten bildet das *Worksheet*-Element, die folgenden Elemente sind *Name*, *Table*, *Column*, *Row*, *Cell* sowie *Data*.

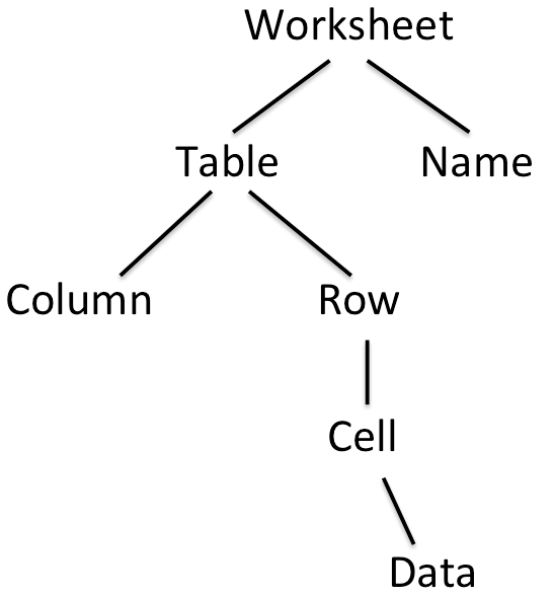


Abbildung 41: Ausschnitt der Baumstruktur des XML-Dokumentes, Excel-Protokolltyp Nr. 10

[33] Als semistrukturierte Daten werden Daten bezeichnet, die keiner allgemeinen Struktur unterliegen

Ein XPath-Ausdruck wird auch als Lokalisierungspfad bezeichnet. Ein Lokalisierungspfad hat die Aufgabe, eine Knotenmenge relativ zu einem Kontextknoten zu bestimmen und kann sich aus mehreren Lokalisierungsschritten zusammensetzen. Jeder Lokalisierungsschritt wählt jeweils eine Knotenmenge des Dokumentes aus. Eine Knotenmenge selbst kann wiederum aus Knoten bestehen [20]. Die Tabelle 8 listet sämtliche Navigationsachsen von XPath auf. Insgesamt existieren 12 Navigationsachsen, die jeweils den Navigationsweg für die Lokalisierung eines Knotens vorgeben.

Achse	**Adressierte Knoten**
child	direkte Nachfolger
descendant	alle Nachfolgerknoten
parent	übergeordnete Elternknoten
attribute	Attributknoten
descendant-or-self	untergeordnete Knoten und Kontextknoten
self	Kontextknoten selbst
following	in Dokumentordnung nachfolgende Knoten
following-sibling	nachfolgende Geschwisterknoten
preceding	in Dokumentordnung vorhergehende Knoten
preceding-sibling	vorhergehende Geschwisterknoten
ancestor	alle Vorgängerknoten
ancestor-or-self	Vorgängerknoten und Knotextknoten
namespace	Namensraumknoten

Tabelle 8: Auflistung der 12 Achsen in XPath

Zu Beginn eines Ausdruckes werden vorab die Kinder und Kindeskinder, ausgehend vom aktuellen Kontextknoten selbst, abgearbeitet. Die einzelnen Lokalisierungsschritte werden mit dem Zeichen *"/"* voneinander getrennt. Ein Lokalisierungsschritt beinhaltet die Angabe der Achse und die Angabe eines Knotens, die jeweils mit dem Zeichen *"::"* voneinander getrennt werden. Die allgemeine Notation für einen Lokalisierungsschritt ist folgendermaßen:

/Achse :: Knoten [Prädikat 1]...[Prädikat n]/... (10)

Jeder Lokalisierungsschritt kann Prädikate enthalten. Die Prädikate werden in eckigen Klammern, direkt nach dem Knoten, angegeben. Mit der Angabe von Prädikaten können weitere Einschränkungen für das Ergebnis festgelegt werden. Weiterhin wird bei XPath zwischen absoluten und relativen Pfadausdrücken unterschieden. Absolute Pfade beginnen immer von der Wurzel eines XML-Dokumentes, relative Pfadausdrücke beginnen vom Kontextknoten, dem aktuellen Knoten. In der vorliegenden Arbeit werden lediglich relative Pfadausdrücke verwendet.
Weiterhin ist zu erwähnen, dass für jeden der zehn Excel-Protokolltypen sogenannte Muster, für die Adressierung der Werte aus den Excel-Zellen, definiert werden müssen. Die definierten Muster beinhalten für jeden zu integrierenden Wert, eine exakte Positionsbestimmung in Form von XPath-Ausdrücken. Mit Hilfe der aufgelisteten XPath-Ausdrücke können die in einer Excel-Zelle befindlichen Werte adressiert werden. Hierzu wurden für jeden Protokolltyp Muster definiert, sodass die Adressierung, der zu jedem Attribut der Datenbank gehören-

den Werte aus den Excel-Eingabeprotokollen, realisiert werden kann. Die Spezifikation der zehn unterschiedlichen Muster, zu den jeweiligen Protokolltypen, ist notwendig, da die zu integrierenden Werte jeweils von Protokolltyp zu Protokolltyp unterschiedliche Positionen aufweisen. In jedem Muster sind jeweils die in der Datenbank beinhalteten Tabellennamen, sowie die dazugehörigen Attribute angegeben. Zu jedem Attribut wurde anschließend ein XPath-Ausdruck definiert. In Abbildung 42 ist ein Ausschnitt eines solchen Musters abgebildet. Die kompletten Muster für Protokolltyp Nr. 1 bis 10 sind dem Anhang zu entnehmen.

Tabelle „Station“

Longitude:
//ss:Worksheet[@ss:Name = 'EingabeMetadaten']/ss:Table/ss:Row/ss:Cell/ss:Data[text() = 'Positionierung (WGS 84):']/parent::*/parent::*/following-sibling::*[1]/ss:Cell/ss:Data[text() = 'E°min,dec']/parent::*/following-sibling::*[2]/ss:Data

Latitude:
//ss:Worksheet[@ss:Name = 'EingabeMetadaten']/ss:Table/ss:Row/ss:Cell/ss:Data[text() = 'Positionierung (WGS 84):']/parent::*/parent::*/following-sibling::*[2]/ss:Cell/ss:Data[text() = 'N°min,dec']/parent::*/following-sibling::*[2]/ss:Data

Name:
//ss:Worksheet[@ss:Name = 'EingabeMetadaten']/ss:Table/ss:Row/ss:Cell/ss:Data[text() = 'Stationsname:']/parent::*/following-sibling::*[1]/ss:Data

Seegebiet_idSeegebiet:
Schlüssel aus Tabelle Seegebiet (idSeegebiet)

Tabelle „Artenliste“

idArtenliste:
auto increment

Name:
//ss:Worksheet[12]/@*

Version:
//ss:Worksheet[@ss:Name = 'EingabeMetadaten']/ss:Table/ss:Row/ss:Cell/ss:Data[text() = 'gültige Artenliste:']/parent::*/following-sibling::*[2]/ss:Data

Abbildung 42: Ausschnitt des Musters für Protokolltyp Nr. 10

Der allgemeine Ablauf des vorliegenden Ansatzes, für die Adressierung der zu integrierenden Werte, ist folgendermaßen:

1. Transformation der Excel-Eingabeprotokolle in das XML-Dateiformat.
2. Muster für die exakte Positionsbestimmung definieren und gleichzeitig die XPath-Ausdrücke aufstellen.
3. Anhand der definierten Muster den zu integrierenden Wert adressieren.

Im Folgenden werden zwei Beispiele erläutert. Beide Beispiele beziehen sich auf den Excel-Protokolltyp Nr. 10.

Beispiel 1

In diesem Beispiel wird gezeigt, wie durch einen XPath-Ausdruck ein Wert aus einem Excel-Eingabeprotokoll adressiert werden kann. In Abbildung 39 ist bereits ein Ausschnitt des Protokolltyps Nr. 10 abgebildet. In dem vorliegenden Beispiel soll aus dem Protokolltyp Nr. 10 der Wert der Temperatur adressiert werden. Die Adressierung des Wertes ist notwendig, um diesen für die spätere Transformation in die Datenbank zur Verfügung zu stellen.
Nachdem das jeweilige Excel-Eingabeprotokoll in das XML-Dateiformat transformiert wurde, kann der Wert anhand der bereits definierten Muster adressiert werden. Die Spezifikation des zugehörigen XPath-Ausdruckes wird im Folgenden erläutert.
Der zu integrierende Wert der Temperatur kann nur in Abhängigkeit eines anderen, in einer Excel-Zelle befindlichen Wertes, adressiert werden. Die Excel-Zelle, die den abhängigen Wert speichert, muss vorab lokalisiert werden. Für die Lokalisierung gilt als Bedingung, dass die im ersten Schritt zu lokalisierende Zelle den String *"Temperatur:"* enthalten muss. Nachdem diese Zelle lokalisiert wurde, kann die eigentliche Excel-Zelle lokalisiert werden, die den zu integrierenden Wert der Temperatur beinhaltet.
Der XPath-Ausdruck konnte anhand des nachfolgenden Ausdruckes 11 spezifiziert werden.

$$\mathbf{C[i,j] = "Temperatur: "} \ \& \ \mathbf{value = C[i, j+1]} \tag{11}$$

Der Ausdruck 11 beinhaltet zum einen die abhängige Zelle mit dem beinhalteten String *"Temperatur:"*. Die Position der Zelle wird jeweils durch einen Spalten- und Zeilenindex bestimmt. Weiterhin beinhaltet der Ausdruck die Zelle, die den zu integrierenden Wert der Temperatur speichert. Die Position der Zelle wird ebenfalls durch einen Spalten- und Zeilenindex bestimmt. Der einzige Unterschied hierbei ist, dass die Position dieser Zelle, die den zu integrierenden Wert beinhaltet, ausgehend von der abhängigen Zelle um einen Spaltenindexwert weiter (j+1) bestimmt wird. Dies wurde bereits in Abbildung 39 illustriert.
Ausgehend von diesen Informationen zur Positionsbestimmung, konnte der dazugehörige XPath-Ausdruck zur direkten Adressierung des Wertes, spezifiziert werden. Der folgende XPath-Ausdruck in Abbildung 43 setzt sich aus acht Lokalisierungsschritten zusammen. Die acht Lokalisierungsschritte sind zusätzlich in Abbildung 43 grafisch dargestellt.

```
//ss:Worksheet[@ss:Name = 'EingabeMetadaten']/ss:Table/ss:Row/ss:Cell
/ss:Data[text() = 'Temperatur (°C):']/parent::*/following-sibling::*[1]/ss:Data
```

Der erste Lokalisierungsschritt besteht darin, das Element *Worksheet* (Tabellenblatt) auszuwählen. In diesem Zusammenhang wurde das Präfix "ss" vor dem Element mit angegeben. Dieses dient der Auswahl des jeweiligen Namensraumknoten (namespace). Das Präfix "ss" steht dabei für Microsoft Office Spreadsheets und muss vor jedem Element in jedem Lokalisierungsschritt angegeben werden.
Zusätzlich muss zur Auswahl des Worksheets ein Prädikat mit angegeben werden, welches die Auswahl des jeweiligen Worksheets eingrenzt. Hierzu wurde der Name des Excel-

Tabellenblattes angegeben, in welchem der zu integrierende Wert gespeichert ist. Anschließend werden über die drei darauf folgenden Lokalisierungsschritte jeweils die Elemente *Table* (Tabelle), *Row* (Zeile) sowie *Cell* (Zelle) ausgewählt. Die Auswahl der Zeile ist damit zu begründen, da das XML-Dokument aus dem Excel-Dokument zeilenweise aufgebaut ist. Der nachfolgende Lokalisierungsschritt wählt das *Data*-Element aus, welches den String "Temperatur" speichert. Diese Auswahl kann durch Angabe des Prädikates [text() = " "] realisiert werden. Mit den bis hierhin durchlaufenen Lokalisierungsschritten wurde die Zelle C[i,j], siehe Ausdruck 11, erreicht.
Über die nachfolgenden Lokalisierungsschritte muss die Zelle C[i,j+1] erreicht werden. Hierzu muss durch die Angabe der Achse *parent*, das Elternelement von *Data* ausgewählt werden. Anhand der zuvor abgebildeten Baumstruktur ist zu erkennen, dass es sich bei dem Elternelement von *Data* hierbei um das Element *Cell* handelt. Von diesem Element aus muss durch einen weiteren Lokalisierungsschritt, durch Angabe der Achse *following-sibling[1]*, die nächste Excel-Zelle (C[i,j+1]) ausgewählt werden. Über den letzten Lokalisierungsschritt, durch die Angabe des *ss:Data*-Elementes, wird der Wert ausgewählt. Das Ergebnis des XPath-Ausdruckes ist der Wert der Temperatur, "9,8" Grad.
Mit Hilfe dieses XPath-Ausdruckes ist es möglich, den zu integrierenden Wert zu adressieren. Weiterhin können durch die Art der XPath-Ausdrücke sämtliche Werte aus einer Excel-Datei adressiert werden.

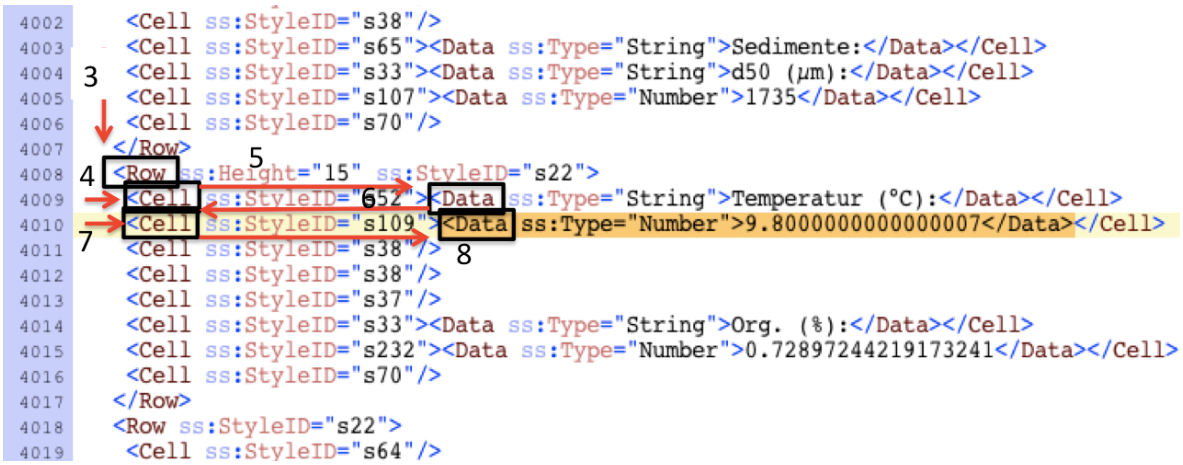


Abbildung 43: Grafische Darstellung der Lokalisierungsschritte aus Beispiel 1

Beispiel 2

Das zweite Beispiel orientiert sich an der Abbildung 44. Mit diesem Beispiel wird eine Adressierung beschrieben, in dem die Lokalisierung des zu integrierenden Wertes von drei in einer Excel-Zelle befindlichen Werte abhängig ist. Gesucht wird der *Bearbeiter Labor* des *Hol 1*.

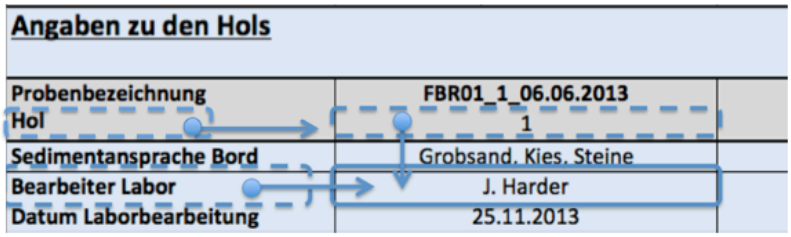


Abbildung 44: Grafische Darstellung der Adressierung des zu integrierenden Wertes über die abhängigen Werte ”Hol”, ”1” und ”Bearbeiter Labor”

Die Werte in den jeweils abhängigen Zellen müssen ermittelt werden, bevor der zu integrierende Wert adressiert werden kann. Der folgende Pfadausdruck zeigt die Adressierung des Wertes.

```
//ss:Worksheet[@ss:Name = 'EingabeMetadaten']/ss:Table/ss:Row/ss:Cell
/ss:Data[text() = 'Hol']/parent::*/parent::*/ss:Cell/ss:Data[text() = '1']
/parent::*/parent::*/following-sibling::*/ss:Cell
/ss:Data[text() = 'Bearbeiter Labor']/parent::*/following-sibling::*[1]/ss:Data
```

Der XPath-Ausdruck beinhaltete 17 Lokalisierungsschritte und beginnt bei dem *Worksheet*-Element. Die Schritte beschreiben das Lokalisieren der Werte ”Hol”, ”1” sowie zuletzt ”Bearbeiter Labor”. Die Achsenangabe *following-sibling* sowie *parent* wurden bereits im vorherigen Beispiel erläutert. Von den Positionen der drei zu lokalisierenden Werte, kann der zu integrierende Wert adressiert werden. Das Ergebnis des XPath-Ausdruckes ist ”J.Harder”.
Durch Weglassen von Lokalisierungsschritten kann der zuvor spezifizierte XPath-Ausdruck optimiert werden. Zum einen können durch das Ersetzen der doppelt angegebenen *parent*-Achsen mit der *ancestor*-Achse, ein Lokalisierungsschritt gespart werden. Somit werden aus vier Lokalisierungsschritte lediglich zwei.
Weiterhin kann die Angabe des Tabellenblattes ignoriert werden und direkt, durch Angabe des Kontextknotens, bei dem Element *Data* für den Wert ”Hol”, begonnen werden. Da in diesem Zusammenhang lediglich die Werte *Hol*, *1* sowie *Bearbeiter Labor* von Interesse sind und alle drei Werte gemeinsam nur auf dem ersten Tabellenblatt des Excel-Eingabeprotokolls vorkommen, kann die Angabe des Worksheets eingespart werden. Ist der Fall gegeben, dass die Werte gemeinsam auf mehreren Tabellenblättern zu finden sind, so muss explizit im ersten Lokalisierungsschritt das jeweilige Worksheet angegeben werden.

Der folgende optimierte XPath-Ausdruck beinhaltet 11 Lokalisierungsschritte. Mit dieser Art des XPath-Ausdruckes können problemlos abhängige Werte adressiert werden. Weitere XPath-Ausdrücke sind dem Anhang zu entnehmen.

```
//ss:Data[text() = 'Hol‘]/ancestor::*/following-sibling::*/ss:Data[text() = '1‘]
/ancestor::*/following-sibling::*/ss:Cell/ss:Data[text() = 'Bearbeiter Labor']
/parent::*/following-sibling::*[1]/ss:Data
```

Zusammenfassung

Der zuvor beschriebene Ansatz ist für die Adressierung der zu integrierenden Werte, aufgrund der Flexibilität, geeignet. Der Vorteil liegt darin, keine eigene beschreibende Sprache spezifizieren zu müssen, sondern auf eine bereits vorhandene Sprache zurückzugreifen. Die XML-Abfragesprache XPath bietet die Möglichkeit, Teile eines XML-Dokumentes zu adressieren. Hierzu werden sämtliche Excel-Eingabeprotokolle in das XML-Dateiformat transformiert. Gleichzeit wird hiermit die Datenmodell-Heterogenität überwunden. XML-Dokumente weisen eindeutige Strukturen auf und erlauben somit eine einfache Verarbeitung der transformierten Excel-Eingabeprotokolle. Ein weiterer Vorteil der Transformation in das XML-Dateiformat ist, dass diese Dokumente mensch- und maschinenlesbare Dokumente darstellen, die für eine breite Palette von Anwendungen sowie plattformunabhägig eingesetzt werden können.
Mit Hilfe von XPath-Ausdrücken können sowohl einzelne Werte, als auch gesamte Bereiche aus Excel adressiert werden. Somit kann die Adressierung einer Menge von Werten realisiert werden.
Weiterhin können durch die zuvor definierten Muster, sämtliche Werte problemlos aus allen Excel-Protokolltypen adressiert werden. Ein weiterer Vorteil der XPath-Ausdrücke liegt darin, dass Prädikate spezifiziert werden können und somit durch relative Pfadausdrücke die Adressierung von abhängigen Werten realisiert werden kann. Dies ist sowohl mit dem ersten, als auch mit dem zweiten Ansatz nicht möglich. Durch die Angabe von Prädikaten kann eine korrekte Adressierung der Werte gewährleistet werden. Können die jeweils angegebenen Bedingungen nicht eingehalten werden, kann kein Wert lokalisiert und somit keine Adressierung durchgeführt werden.

Im weiteren Verlauf der Arbeit wird der 3. Ansatz verfolgt. Der 1. und 2. Ansatz eignen sich nicht für die Adressierung der Werte aus den Excel-Eingabeprotokollen. Dies wurde bereits begründet.
Der nächste Schritt im Schema Mapping Prozess ist die Generierung der Transformationsanfragen.

5.2.3. Transformationsanfragen

Der dritte Schritt im Schema-Mapping Prozess besteht darin, Transformationsvorschriften aus den zuvor spezifizierten Mappings abzuleiten. Die Transformationsvorschriften werden durch eine Anfragesprache wie SQL oder XQuery beschrieben. Das Ziel der Transformationsanfrage ist die Verknüpfung des Quell- und Zielschemas durch eine konkrete Anfrage. In diesem Zusammenhang müssen alle zuvor aufgestellten Wertkorrespondenzen beachtet werden. Dieser Prozessschritt wird auch als *Interpretation des logischen Mappings* bezeichnet [24, S. 126].
Der dritte Prozessschritt ist in Abbildung 45 dargestellt. Mit Hilfe der Transformationsanfrage werden die konkreten Daten des Quellschemas in die Struktur des Zielschemas überführt [24, S. 125].

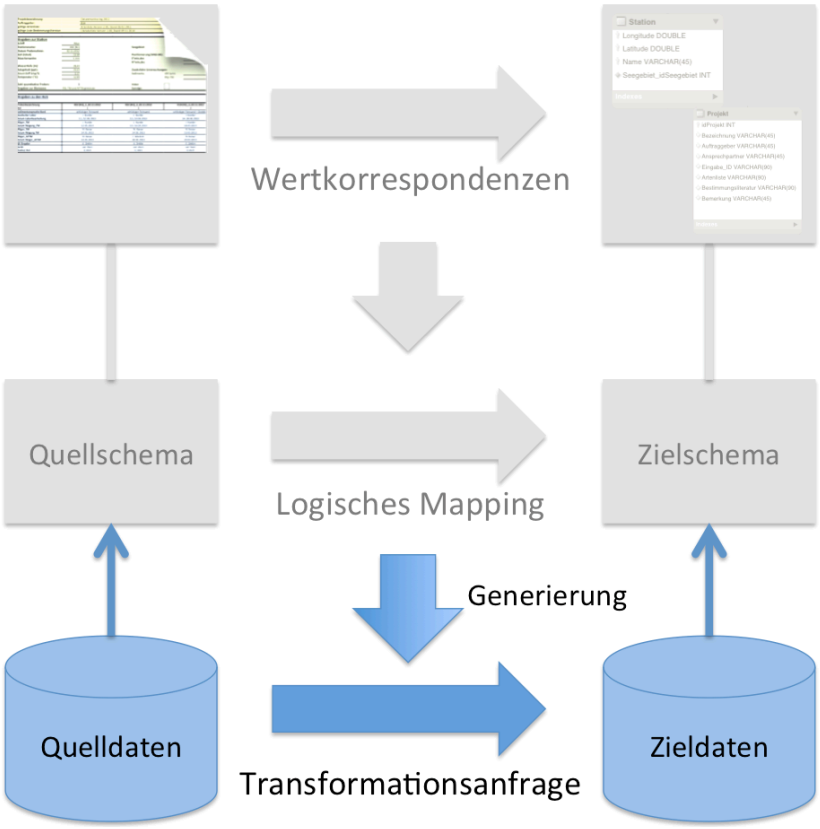


Abbildung 45: Ableiten der Transformationsanfrage

Eine Möglichkeit die Transformation zu beschreiben, ist eine SQL-Anweisung, z.B eine *INSERT INTO* oder eine *SELECT FROM WHERE* SQL-Anfrage [25].
Für jedes erzeugte logische Mapping muss eine Anfrage generiert werden. Das Listing 2 zeigt eine allgemeine Anfrage für die Transformation der in den Quellschema enthaltenen Daten in die jeweilige Relation des Zielschemas, in Form einer abgeleiteten SQL-Anfrage.

```
1 CREATE    VIEW name AS
2 SELECT    Excel.Attribut AS DB.Atrtibut, ...
3 FROM      Excel.Tabelle
```

Listing 2: Allgemeine Transformationsanfrage

Voraussetzung für eine Transformationanfrage sind zum einen das Quellschema und das Zielschema und zum anderen eine Menge von Korrespondenzen zwischen den Schemata selbst. Eine Transformationsanfrage muss bestimmte Eigenschaften erfüllen. Zum einen darf die Semantik des Quellschemas, in diesem Fall die Semantik der beinhalteten Daten des Quellschemas, nicht verändert werden. Zum anderen müssen die Schlüssel- sowie Integritätsbedingungen des Quell- und Zielschemas sowie die spezifizierten Korrespondenzen beachtet werden [24, 34].

In diesem Abschnitt wird das Ableiten einer Transformationsanfrage anhand eines Beispiels beschrieben. Die konkrete Umsetzung dieser Anfragen wird in Kapitel *Implementierung* vorgenommen.
Die Eingabe für die Transformationsanfragen sind die aus 5.2.1 spezifizierte Menge von Korrespondenzen zwischen den Schemata sowie die Quellschemata und das Zielschema. Die unterschiedlichen Quellschemata sind die zur Verfügung stehenden Eingabeprotokolle und das Zielschema, welches bereits in Abschnitt 5.1 modelliert wurde.

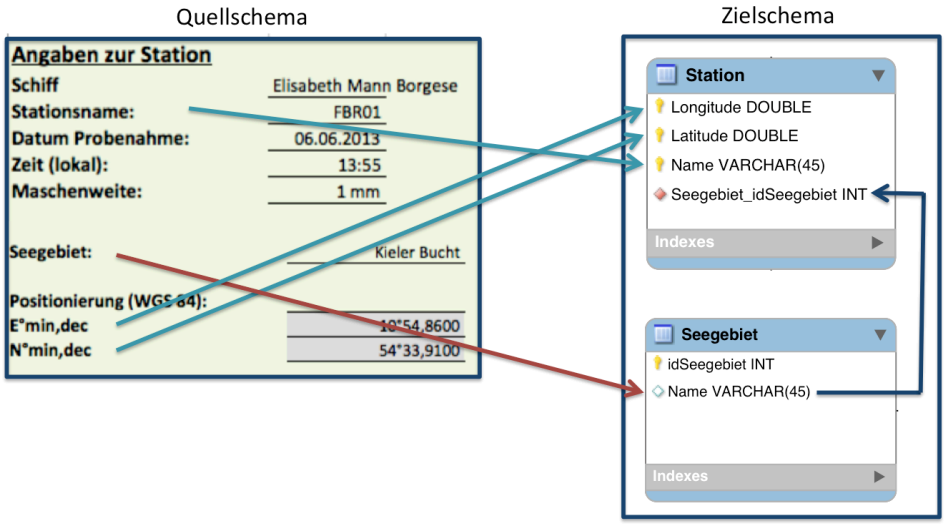


Abbildung 46: Wertkorrespondenzen zwischen Quell- und Zielschema bezüglich der Tabellen ”Station” und ”Seegebiet”, die in Beziehung zueinander stehen; Excel-Protokolltyp Nr. 10

Im Folgenden wird die Transformation anhand eines konkreten Beispiels erläutert. Die Abbildung 46 illustriert die Wertkorrespondenzen zwischen dem Quellschema (Excel-Protokolltyp Nr. 10) und den Tabellen *Station* und *Seegebiet*, die Teil des Zielschemas sind. Weiterhin ist die Beziehung zwischen den beiden Relationen über den Fremdschlüssel *Seegebiet_idSeegebiet* abgebildet.

Das Ziel der Anfrage ist die Transformation der Daten aus den Attributen des Quellschemas in die Zieldaten der Attribute *Longitude*, *Latitude* sowie *Name* der Datenbanktabelle *Station*. Um die Daten aus den Eingabeprotokollen in die Tabelle Station transformieren zu können, müssen vorab die Daten aus den Eingabeprotokollen der Tabelle Seegebiet transformiert werden. Dies ist notwendig, da die Transformation die Schlüssel- sowie Integritätsbedingungen des Zielschemas beachten muss. Um die Daten in die Relation Station integrieren zu können, muss der Fremdschlüssel *Seegebiet_idSeegebiet* beachtet werden, der in der Tabelle Seegebiet enthalten ist.

Die in der Abbildung 46 gezeigten Attribute des Quellschemas korrespondieren mit jeweils einem Attribut der Zieltabelle [24, 34]. Eine Interpretation der Mappings, welche die Fremdschlüsselbeziehung zwischen den Zielrelationen beachtet, leitet nachfolgende Anfrage zur Transformation ab:

```
1  CREATE  VIEW Seegebiet AS
2  SELECT   Seegebiet AS Name
3  FROM     Excel
4
5  CREATE  VIEW Station AS
6  SELECT   Stationsname AS Name, Emin,dec AS Longitude, Nmin AS Latitude
7  FROM     Excel
8  WHERE   Seegebiet\_idSeegebiet
9                (SELECT idSeegebiet FROM Seegebiet
10               WHERE Seegebiet.Excel = Seegebiet.Name)
```

Listing 3: Konkretes Beispiel

In der ersten Anfrage im Listing 3 werden die Daten bezüglich des Seegebietes transformiert, um darauf aufbauend in der zweiten Anfrage die Daten bezüglich der Station zu transformieren. Für die zweite Anfrage muss der Schlüssel aus der Tabelle Seegebiet abgefragt werden und mit dem Namen des Seegebietes aus den Excel-Eingabeprotokollen verglichen werden. Die in Listing 3 spezifizierte Transformationsanfrage beachtet die Struktur der beiden Schemata. Die Integritätsbedingung des Zielschemas wird nicht verletzt und alle in 46 abgebildeten Korrespondenzen werden berücksichtigt. Das Ergebnis der Spezifikation ist eine formale Interpretation des Mappings mit den vier enthaltenen Wertkorrespondenzen sowie der Beziehung zwischen den beiden Tabellen des Zielschemas. Das Ergebnis spiegelt eine Intention des Anwenders wieder.
Die Anfrage ist eine theoretische Lösungsvariante für die Transformation der Daten aus den Excel-Eingabeprotokollen in die Datenbank. Die Transformation der Daten aus den Eingabeprotokollen kann jedoch nicht einfach mit einer SQL-Anfrage realisiert werden, da die Datensätze direkt in einer Excel-Tabelle gespeichert sind und nicht in anderen Datenbanktabellen. Hierzu müssen die bereits beschrieben Lösungsansätze für die Adressierung der Daten beachtet werden. Eine konkrete Umsetzung der Transformation, die die Adressierung aus Excel beachtet, wird in Kapitel 6 vorgestellt. Hierzu wird zum einen eine einfache *Insert Into* SQL-Anweisung und zum anderen eine *Load Data* Anweisung umgesetzt.

5.3. Zusammenfassung

In diesem Kapitel wurde ein Konzept für die Integration der in den Excel-Eingabeprotokollen gespeicherten Daten vorgestellt. Hierzu wurde in Abschnitt 5.1 das Zielschema für die Datenintegration vorgestellt. Das konkrete Datenbankkonzept wurde mit dem grafischen Benutzerwerk MySQL Workbench realisiert.
Im zweiten Teil des Kapitels wurde ein konkretes Konzept für die Datenintegration beschrieben. Das Integrationskonzept dieser Arbeit wurde auf Basis des Schema Mapping Prozesses, welcher in Kapitel 3 erläutert wurde, entworfen. Hierzu wurden zu Beginn Wertkorrespondenzen zwischen den Excel-Protokolltypen und dem Zielschema spezifiziert. Mit dem Konzept sollen möglichst alle Daten der Excel-Eingabeprotokolle in die Struktur des Zielschemas überführt werden. Anschließend wurden drei Ansätze vorgestellt, mit deren Hilfe die Adressierung der Daten aus den Excel-Eingabeprotokollen ermöglicht werden kann. Im weiteren Verlauf der vorliegenden Arbeit wird der dritte Ansatz weiter verfolgt. Dieser Ansatz beschreibt die Adressierung über XPath-Ausdrücke, mit deren Hilfe die Werte in den Excel-Dokumenten lokalisiert werden können. Im letzten Schritt des Integrationskonzeptes, Abschnitt 5.2.3 wurden die Transformationsanfragen vorgestellt, mit deren Hilfe die konkreten Daten der Excel-Eingabeprotokolle in die Struktur des Zielschemas überführt werden.

6. Implementierung

In diesem Kapitel wird die Umsetzung der Datenintegration beschrieben. Hierzu werden die festgelegten Abläufe und Strukturen, die in Kapitel 5 vorgestellt wurden, umgesetzt. Das Datenbankmodell wurde mit Hilfe der *MySQL Workbench* erstellt und bereits in 5.1 vorgestellt. Die Generierung der Datenbanktabellen konnte im Wesentlichen automatisch erfolgen. Den Schwerpunkt dieses Kapitels bildet die Umsetzung der Datenintegration.
In diesem Kapitel werden in Abschnitt 6.1 drei Implementierungsansätze vorgestellt und diskutiert, mit deren Hilfe die Adressierung der Werte aus den Excel-Protokollen sowie die Transformation der adressierten Daten umgesetzt werden kann. Lösungsansätze für die Adressierung der in Excel gespeicherten Daten wurden bereits in Kapitel 5 vorgestellt. Anschließend erfolgt in Abschnitt 6.1.4 ein Vergleich dieser Ansätze. In Abschnitt 6.2 wird die Umsetzung der Datenintegration zusammengefasst.

6.1. Verwendete Implementierungsansätze

Für die Implementierung stehen drei verschiedene Ansätze zur Auswahl. Die Werte können zum einen direkt über eine Datei im Comma-separated values (CSV)-Format in die Datenbank transformiert werden, zum anderen können die Daten durch ein spezielles Tool von Excel in eine MySQL Datenbank importiert werden. Außerdem besteht die Möglichkeit, über die Skriptsprache *PHP* die Adressierung der Werte zu realisieren. Für die Umsetzung werden im Folgenden die drei verschiedenen Implementierungsansätze getestet sowie Vor- und Nachteile diskutiert.

6.1.1. CSV-Dateiformat

Excel-Tabellen lassen sich problemlos im CSV-Dateiformat abspeichern. Die Eigenschaft des Formates ist die Speicherung von Daten durch ein Trennzeichen. Das Trennzeichen ist meist ein Semikolon oder ein Komma.

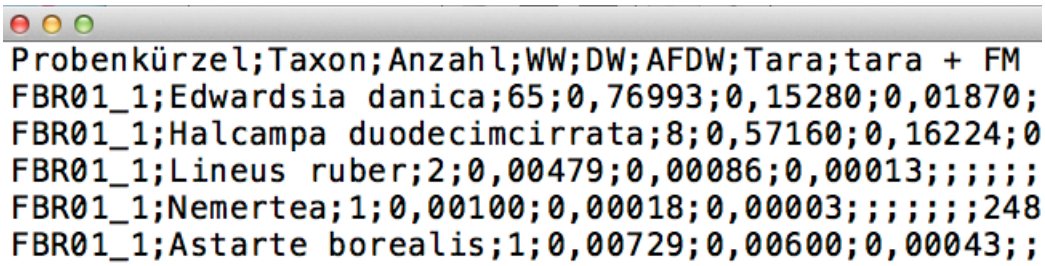


Abbildung 47: Ausschnitt eines Excel-Eingabeprotokolls im CSV-Dateiformat

Die Abbildung 47 zeigt den Ausschnitt einer CSV-Datei. Jede Zeile in einer CSV-Datei entspricht einem Tupel[34] in der Datenbank. Die Daten sind hierbei durch ein Semikolon getrennt.

[34]Ein Tupel ist eine Zeile (Reihe) in einer Datenbank-Tabelle

Für den Import der Datensätze müssen sämtliche Excel-Eingabeprotokolle in das CSV-Dateiformat konvertiert werden. Um die im CSV-Dateiformat gespeicherten Daten in die Datenbank zu speichern, wird eine einfache Anweisung im SQL-Editor der MySQL Workbench eingegeben, mit deren Hilfe die Daten in die Datenbank transformiert werden können. Das Listing 4 zeigt den Aufbau einer *LOAD DATA LOCAL INFILE*-Anweisung. Ein konkretes Beispiel hierfür wird im nächsten Abschnitt erläutert.
Mit Hilfe der Anweisung aus Listing 4 können die Datensätze der CSV-Datei in die Datenbank importiert werden. Über (Zeile 1, Listing 4) wird der Datenbank mitgeteilt, dass eine lokale CSV-Datei zur Verfügung steht. Der Pfad der Datei muss mit angegeben werden. Im zweiten Schritt (Zeile 2, Listing 4) wird angegeben, in welche Datenbank-Tabelle die zu importierenden Daten gespeichert werden sollen. Die Anweisungen in (Zeile 3 & 4, Listig 4) sind dafür zuständig der Datenbank mitzuteilen, dass die Datensätze über ein Semikolon getrennt gespeichert sind, dass die Datensätze auch durch Anführungsstriche eingeschlossen werden können und das jede Zeile in der CSV-Datei mit einen Zeilenumbruch beendet wird. Zuletzt müssen die Attributnamen der CSV-Datei aufgelistet werden (Zeile 5, Listing 4). Anhand dieser Auflistung wird die Reihenfolge der zu speichernden Daten in die Datenbanktabelle festgelegt.

```
1 LOAD DATA LOCAL INFILE 'file_name.csv'
2 INTO TABLE table_name
3 fields terminated by ';' enclodes by ' '' '
4 lines terminated by '\r' ignore 1 lines
5 (attribute_names from table_csv);
```

Listing 4: Load Data Local Infile Anweisung

Konkretes Beispiel

Im Folgenden wird ein konkretes Beispiel erläutert. In diesem Beispiel soll die Tabelle *Art* der Datenbank mit Daten gefüllt wird, siehe hierzu Abbildung 48.

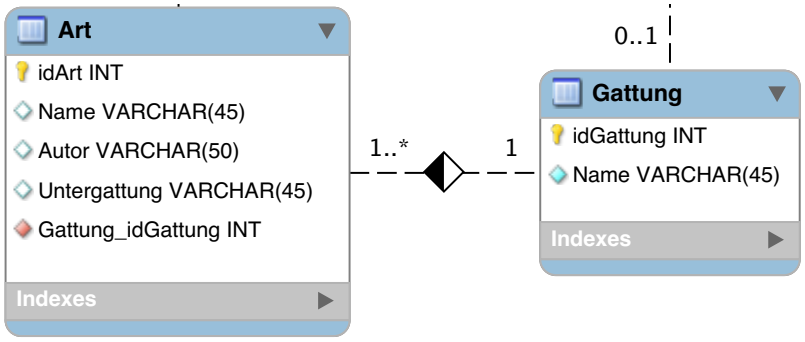


Abbildung 48: Tabelle "Art" mit Beziehung zur Tabelle "Gattung"

Um die Daten in die Tabelle *Art* importieren zu können, wird zunächst die Artenliste, die in dem 12. Tabellenblatt des Excel-Protokolltyps Nr. 10 gespeichert ist, als CSV-Datei gespeichert. Die CSV-Datei steht anschließend für den Import der Datensätze zur Verfügung. Das Listing 5 zeigt eine komplette *LOAD DATA LOCAL INFILE* Datenbank-Anweisung, um die Datensätze in die Tabelle *Art* zu importieren.

```
1 LOAD DATA LOCAL INFILE '.../.../Artenliste.csv'
2 INTO TABLE Art
3 fields terminated by ';' enclodes by ' '' '
4 lines terminated by '\r' ignore 1 lines
5 (@dummy, Autor, @dummy, @dummy, @gattunginit, Name, @dummy, @dummy)
6 SET Gattung_idGattung =
7 (SELECT idGattung FROM Gattung WHERE @gattunginit = Name);
```

Listing 5: Beispiel für Load Data Local Infile Anweisung mit eingeschlossener SQL-Anfrage

Um die Anweisung ausführen zu können, muss vorausgesetzt werden, dass in der Tabelle *Gattung* Datensätze gespeichert sind. Dies ist auf die Fremdschlüsselbeziehung zwischen *Art* und *Gattung* zurückzuführen. Die Anweisung wird im Folgenden erläutert.
In jeder Zeile der generierten CSV-Datei finden sich jeweils 8 Werte, die über ein Semikolon voneinander getrennt sind. Die Werte müssen in der (Zeile 5, Listing 5) angegeben werden. Der Name *@dummy* hat keine weitere Bedeutung. Über den Namen wird lediglich angegeben, dass die an dieser Position stehenden Werte der CSV-Datei nicht in die Tabelle der Datenbank importiert werden sollen. Wird die konkrete Bezeichnung eines Attributes der Datenbank-Tabelle angegeben, so werden sämtliche an dieser Position stehenden Werte in die Spalte des jeweiligen Attributes importiert. Mit der Angabe von *@gattunginit* werden die an dieser Position stehenden Werte, der Name der Gattung ebenfalls übernommen. Über (Zeile 6 & 7, Listing 5) wird die Fremdschlüsselbeziehung realisiert. In der Tabelle *Art* wird der Schlüssel *idGattung* aus der Tabelle *Gattung* benötigt. In der CSV-Datei sind ebenfalls

sämtliche Namen der Gattung aufgelistet, die bereits durch die Angabe von *@gattunginit* gemerkt wurden. Über (Zeile 7, Listing 5) erfolgt eine Abfrage des Schlüssels aus Tabelle *Gattung*. Um den jeweils korrekten Wert des Schlüssels zu erhalten, muss der Name der Gattung aus der CSV-Datei gleich dem Gattungsnamen aus der Datenbank-Tabelle *Gattung* sein. Dieser Wert wird in die Tabelle *Art* als Fremdschlüssel gespeichert.

Das universelle CSV-Dateiformat ist eines der ältesten Datenaustauschformate und kann von den meisten Softwareprodukten unter allen Betriebssystemen verwendet werden. Daten, die im CSV-Dateiformat gespeichert sind können, aufgrund der einfachen Struktur von allen Anwendern verstanden und bearbeitet werden. Diese genannten Vorteile stellen auch gleichzeitig einen Nachteil dar, denn aufgrund der Einfachheit können komplexe Datenstrukturen und Abhängigkeiten nicht abgebildet werden [35]. Da jedoch die meisten in Excel gespeicherten Daten nicht strukturiert vorliegen, bietet die Transformation in eine CSV-Datei keinen guten Lösungsansatz für die Integration der Daten aus Excel. Soll eine Excel-Tabelle mit unstrukturierten Daten in die Datenbank integriert werden, dann ist ein anderer Implementierungsansatz zu wählen. Die im CSV-Dateiformat gespeicherten Werte können nicht über die LOAD DATA LOCAL INFILE Anweisung importiert werden.
Ein Vorteil ist, dass über diese Anweisung die Datensätze mit einer hohen Geschwindigkeit aus einer Datei in eine Datenbank-Tabelle eingelesen werden. Aus diesem Grund wurde dieser Ansatz für die Implementierung herangezogen [1]. Weiterhin ist die Anweisung einfach zu definieren und funktioniert bei einfachen Tabellenstrukturen. Diese strukturierten Daten lassen sich problemlos in eine CSV-Datei abspeichern und anschließend importieren. Ein weiterer Vorteil ist die einfache Realisierung der Fremdschlüsselbeziehung über eine weitere SQL Abfrage.

6.1.2. MySQL for Excel

Das Tool **MySQL for Excel** für Microsoft Windows bietet eine einfache Lösung, um Daten aus einer Excel-Tabelle in eine MySQL Datenbank zu importieren. Das Tool wird automatisch mit dem MySQL Installer installiert und ist über das Tabellenkalkulationsprogramm Excel erreichbar [3]. Im weiteren Abschnitt wird das Tool für Microsoft Excel vorgestellt sowie die Integration der Quelldaten in das Zielschema beschrieben. In diesem Zusammenhang werden sowohl Vor-, als auch Nachteile diskutiert.

Abbildung 49: Grafische Oberfläche des ”MySQL for Excel”-Tools

Die Abbildung 49 zeigt die grafische Oberfläche des Tools, welches direkt über das Programm Microsoft Excel zu bedienen ist. Das Tool bietet die Funktionalitäten, Daten sowohl in eine bestehende Datenbank, als auch in eine über das Tool neu angelegte Datenbank-Tabelle zu exportieren und Daten aus einer Datenbank in Excel zu importieren. Zusätzlich können die in MySQL gespeicherten Daten direkt aus Excel heraus bearbeitet werden. Das Tool bietet die Möglichkeit, die unterschiedlichen Tabellen einer Datenbank auszuwählen, um konkret bestimmen zu können, in welcher Tabelle die Daten gespeichert werden sollen.
Über das Tool lassen sich unstrukturierte Daten importieren. Um die Datensätze aus den Excel-Tabellen zu integrieren, muss zunächst eine Verbindung zur Datenbank hergestellt werden. Anschließend können die in dem Datenbankschema vorliegenden Tabellen ausgewählt werden. Die Zuordnung von Excel-Spalten auf eine Spalte der Datenbank-Tabelle kann automatisch oder manuell erfolgen. Bei der automatischen Zuordnung werden die Spalten aus Excel in der Reihenfolge zugeordnet, in der diese markiert wurden. Bei der manuellen Zuordnung hingegen kann über das Tool die Reihenfolge der zu importierenden Spalten selbst festgelegt werden. Somit können die Spalten beliebig ausgewählt werden. Weiterhin können eine Reihe an Excel-Spalten markiert und in die ausgewählte Tabelle des Datenbankschemas importiert werden. Die Datensätze lassen sich problemlos übernehmen. Die

Abbildung 50 zeigt einen Ausschnitt der Spaltenzuordnung über das Tool. Hierzu wurde die Excel-Spalte *Anzahl Individuen* für den Import ausgewählt. Das abgebildete Dialogfeld zeigt die manuelle Zuordnung der ausgewählten Spalte der Datenbank-Tabelle.

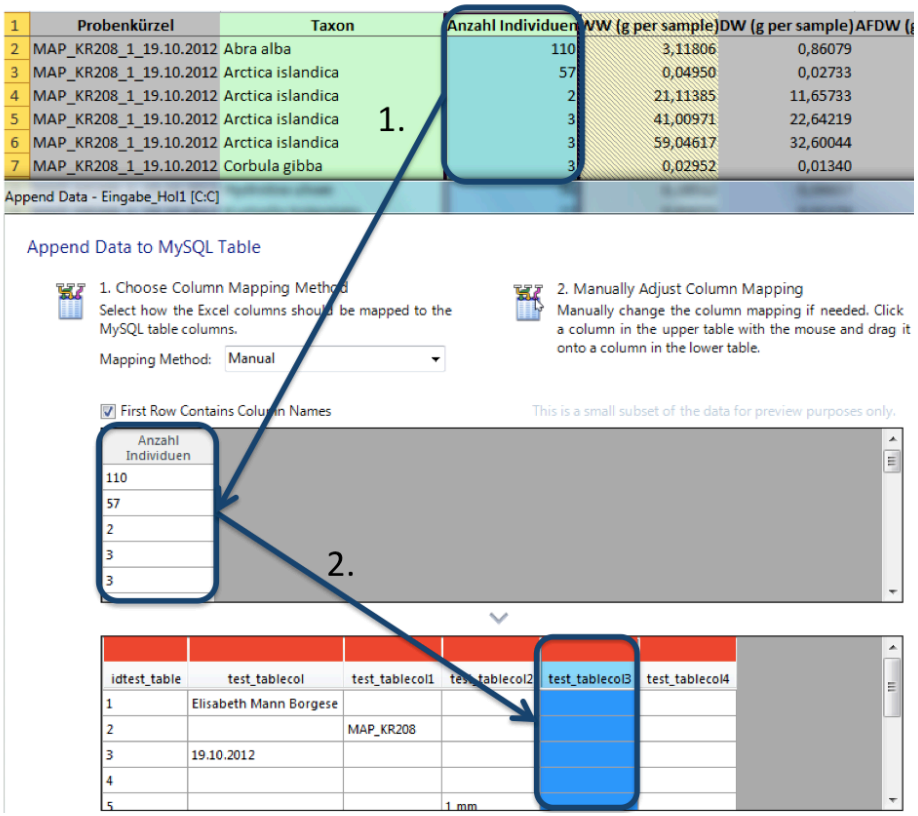


Abbildung 50: Zuordnung einer Excel-Spalte auf Spalte der Datenbank-Tabelle

Das Tool bietet eine Reihe weiterer Einstellungen für den Im- und Export. Die Daten können direkt aus Excel in neu angelegte Datenbankschemata exportiert werden. Hierzu können Einstellungen wie Schlüssel, Unique Index, Namen sowie Datentyp der Attribute festgelegt werden.

Ein wesentlicher Unterschied zum vorherigen Implementierungsansatz liegt in der Auswahl einzelner Zellen aus Excel. Der Wert der Zelle kann anschließend in die Datenbank gespeichert werden. Hierbei kann exakt bestimmt werden, welcher Wert aus Excel zu welchem Attribut der Datenbank-Tabelle zugeordnet werden muss. Das Problem in der Auswahl einzelner Excel-Zelle ist, dass diese nicht zu einem Tupel in der Datenbank zusammengefasst werden können. Ein Beispiel ist die Tabelle *Station*. Hierbei müssen sowohl der Name, als auch die Positionierung gespeichert werden. Diese Werte können zwar über das Tool gemeinsam ausgewählt werden, jedoch nicht gemeinsam in die Datenbank-Tabelle importiert werden, siehe hierzu Abbildung 51.

Um dieses Problem zu lösen, könnten die drei Zellen mit den zu importierenden Werten nacheinander ausgewählt und direkt in die Datenbank-Tabelle gespeichert werden. Das Problem ist jedoch, dass die nacheinander zu importierenden Daten anschließend nicht zu einem Tupel in der Datenbank-Tabelle zusammen geführt werden können. Die Werte können nur getrennt voneinander, in diesem Fall als drei Tupel, in der Datenbank abgelegt werden.

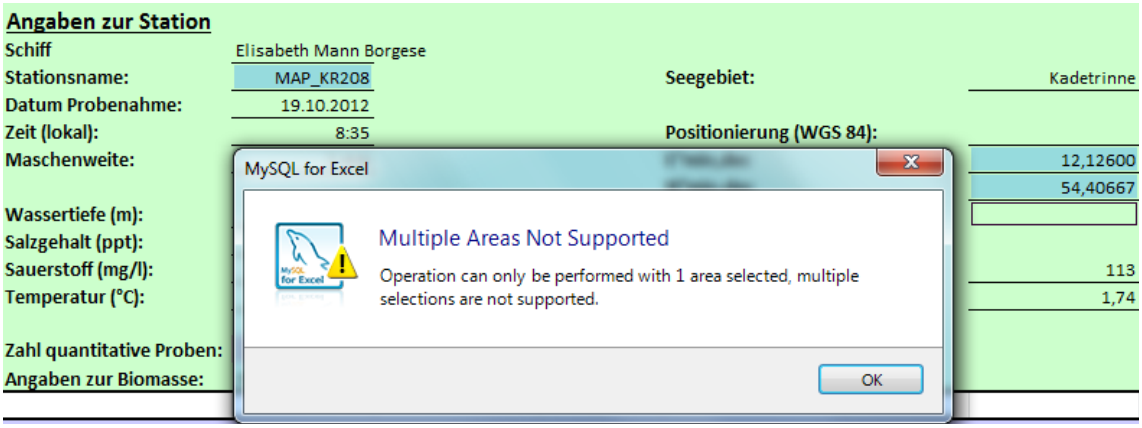


Abbildung 51: Keine gemeinsame Auswahl mehrere Zellen für den Import möglich

Zusammenfassend ist zu sagen, dass dieser Implementierungsansatz eine flexiblere Lösung im Gegensatz zum ersten Ansatz darstellt, da auch einzelne Werte ausgewählt und integriert werden können. Ein weiterer Vorteil ist die relativ einfache Handhabung. Auf der einen Seite sind lediglich die Excel-Spalte auszuwählen und auf der anderen Seite können problemlos die Tabellen und deren Attribute des ausgewählten Datenbankschemas selektiert werden. Der Import erfordert keine Einarbeitungszeit und kann zudem ohne theoretisches Datenbankwissen bedient werden. Weiterhin ist keine Implementierung aufwändiger SQL-Anweisungen notwendig. Der Import wird automatisch durch den Anwender über die grafische Oberfläche gelöst.
Zusammenfassend ist zu sagen, dass das Tool nicht für die Integration der Quelldaten geeignet ist, da keine Zusammenführung von einzelnen ausgewählten Werten möglich ist.

6.1.3. PHP

Das DBMS MySQL kann mit der Open-Source-Skriptsprache PHP verwendet werden und bietet zudem einen umfangreichen Katalog an Funktionen an. Ein wesentlicher Vorteil liegt darin, dass PHP eine Menge von Datenbanken wie MySQL unterstützt [4]. Dieser Vorteil wird ausgenutzt und als weiteren Implementierungsansatz vorgestellt.
Es existieren zwei Möglichkeiten, um Daten in einer Excel-Zelle mit PHP zu adressieren und automatisch in die Datenbank zu transformieren. Die beiden Möglichkeiten werden im Folgenden erläutert.
Die grundlegende Funktionsweise und Syntax von PHP wird in der vorliegenden Arbeit nicht erläutert. An dieser Stelle wird auf die offizielle Dokumentation von PHP in [5] verwiesen.

1. Möglichkeit

PHP unterstützt Datenbanken wie MySQL. In diesem Abschnitt wird die Adressierung über eine absolute Angabe der Position einer Excel-Zellen beschrieben. Für diese Realisierung wird eine schon vorhandene Bibliothek namens *PHPExcel* für das Auslesen von Excel-Zellen herangezogen. Diese Bibliothek kann kostenlos heruntergeladen werden und muss in das PHP-Programm eingebunden werden [7].
Das Listing 6 zeigt einen PHP-Code-Ausschnitt, über den die Adressierung sowie die anschließende Transformation der Daten über eine SQL-Anfrage möglich ist. Der komplette Code für dieses Beispiel ist dem Anhang B zu entnehmen.
Der erste Schritt besteht darin, eine Verbindung zur MySQL Datenbank herzustellen. Die Verbindung wird über die *mysql_connect*-Funktion hergestellt. Die eingebundene Bibliothek in (Zeile 8, Listing 6) ermöglicht die Verbindung zu einer Excel-Datei, die explizit angegeben werden muss. Die Bibliothek ermöglich über eine Reihe an vordefinierten Funktion das Auslesen von Werten aus Excel-Zellen. Um die Daten aus Excel in die Datenbank zu transformieren, wird eine SQL *INSERT INTO* Anfrage generiert. Diese Anfrage ist für jede Tabelle der Datenbank zu spezifizieren. Innerhalb der Anfrage, in der Zuweisung der Attribute (Zeile 24, Listing 6), werden die absoluten Positionen der Excel-Zellen (Spalte und Zeile) angegeben. Die Position wird über den Ausdruck *sheets[0][‘cells‘][Zeile][Spalte]* spezifiziert. Dabei wird über *[0]* das erste Tabellenblatt ausgewählt. Hierbei wird nicht von 1 sondern von 0 hochgezählt.
Um eine Fremdschlüsselbeziehung zu realisieren, muss eine weitere SQL-Anfrage generiert werden. In (Zeile 32 bis 37, Listing 6) wird eine solche Abfrage spezifiziert. Das Ergebnis dieser Anfrage muss gespeichert werden, da das Ergebnis in einer anderen SQL-Anfrage benötigt wird.
Hierbei ist es notwendig vorab sämtliche Tabellen mit Daten zu füllen, in der ein Schlüssel in einer anderen Tabelle als Fremdschlüssel fungiert.

```
1 // PHPExcel-Bibliothek einbinden
2 require_once 'reader.php';
3
4 // neues Excel Objekt generieren; aus
5 // Bibliothek
6 $objExcel = new Spreadsheet_Excel_Reader();
7
8 // Name der Excel-Datei angeben --> in .xls Format bringen
9 $objExcel->read('....xls');
10
11 $i = $objExcel->sheets[0]['numRows'];
12
13 // SQL INSERT Anweisung; Tabelle SEEGEBIET füllen
14     $strQuery = "INSERT INTO `Seegebiet`
15
16 // Tabelle STATION füllen, Fremdschlüssel beachten
17 // 1.   Anfrage nach Fremdschlüssel `Seegebiet_idSeegebiet
18 $query = "SELECT idSeegebiet
19                 FROM Seegebiet
20                 WHERE Name = '" . $objExcel->
21                                 sheets[0]['cells'][8][7] . "'";
22 $result = mysql_query($query);
23 $row = mysql_fetch_assoc($result);
24
25 // 2.   SQL Anfrage generieren
26 // SQL INSERT Anweisung; Tabelle STATION füllen
27     $strQuery = "INSERT INTO `Station` SET
28     `Longitude` = '" . $objExcel->sheets[0]['cells'][11][7] . "',
29     `Latitude` = '" . $objExcel->sheets[0]['cells'][12][7] . "',
30     `Name` = '" . $objExcel->sheets[0]['cells'][8][2] . "',
31     // auf Fremdschlüssel referenzieren
32     // Ergebnis aus Anfrage nutzen
33     `Seegebiet_idSeegebiet` = '" . $row['idSeegebiet'] . "'";
```

Listing 6: Konkretes Beispiel für die Adressierung über PHPExcel-Bibliothek

Der Lösungsansatz eignet sich nicht für die Adressierung der zu integrierenden Werte, da die Positionen der Werte nicht absolut angegeben werden kann. Soll dieser Ansatz verfolgt werden, dann muss für jedes der 10 Protokolltypen ein komplettes PHP-Skript geschrieben werden. Dazu müssen sämtliche Positionen der zu integrierenden Werte festgehalten werden, um diese anschließend im PHP-Skript anzugeben. Über diesen Lösungsansatz ist eine Fehlerbehandlung unmöglich. An dieser Stelle muss zu jeder Zeit gewährleistet werden, dass die Werte auch an den angegebenen Positionen im Excel-Dokument gespeichert sind. Ist ein Wert durch einen Fehler an eine andere Position verrutscht, so werden fehlerhafte Werte transformiert. Die absolute Adressierung wird in der vorliegenden Arbeit nicht verwendet. Einen flexibleren Implementierungsansatz stellt die Adressierung über XPath in PHP dar.

2. Möglichkeit

Die Adressierung der zu integrierenden Quelldaten aus den Excel-Zellen wird über spezifische XPath Ausdrücke realisiert.
In diesem Abschnitt soll die Adressierung der zu integrierenden Informationen aus den Excel-Protokollen anhand des 3. Ansatzes aus Abschnitt 5.2.2 umgesetzt werden.
PHP unterstützt XML und XPath. Somit können über PHP, XPath-Abfragen auf XML-Dokumente spezifiziert und ausgeführt werden. Diese Eigenschaft allein reicht, um den 3. Ansatz für die Adressierung mit PHP umzusetzen. Im Folgenden wird anhand von Programmcode-Ausschnitten die Implementierung der XPath-Ausdrücke beschrieben.
Das Listing 7 zeigt einen PHP-Code-Ausschnitt für die Adressierung der zu integrierenden Werte.

```
1 <?php
2
3 // Laden der XML Datei
4 $xml = simplexml_load_file("....xml");
5
6 // Generieren des XPath-Ausdruckes
7 // Bsp. um den Wert ''Stationsname'' zu
8 // adressieren
9 $result = $xml->xpath(" //ss:Worksheet[@ss:Name =
10 'EingabeMetadaten']/ss:Table/ss:Row/ss:Cell/
11 ss:Data[text() = 'Stationsname:']/parent::*/
12 following-sibling::*[1]/ss:Data ");
13
14  ?>
```

Listing 7: Konkretes Beispiel für die Adressierung über XPath-Ausdrücke

Zunächst wird das XML-Dokument über *simplexml_load_file* geladen. Anschließend werden die XPath-Ausdrücke über *$xml -> xpath(" ausdruck ")* spezifiziert. Anschließend müssen die über XPath adressierten Werte gespeichert werden. Hierbei ist sinnvoll, die Werte in eine CSV-Datei abzulegen. Der Umgang mit CSV-Dateien sowie die Transformation der beinhalteten Daten wurde bereits in 6.1.1 erläutert.

Das Listing 8 zeigt einen weiteren PHP-Code-Ausschnitt, der die adressierten Werte über XPath in eine CSV-Datei speichert. Hierzu wird die von PHP zur Verfügung stehende Funktion *fputcsv()* genutzt, mit deren Hilfe die Werte in eine CSV-Datei abgelegt werden. *$daten* beinhaltet die Daten, die die CSV-Datei speichern soll, in diesem Fall die über XPath adressierten Werte. Die Daten müssen anschließend über eine Transformationsvorschrift in die Datenbank transformiert werden.

```
1 <?php
2 // Öffnen eines Datei-Streams
3 // Angabe des Dateinamens
4 // Datei zum Schreiben öffnen
5 $fp = fopen('Dateiname.csv', 'w')
6
7 // Werte in CSV Datei ablegen
8 $fputcsv($fp, $daten);
9
10 // Schließen der Datei
11 fclose($fp);
12
13  ?>
```

Listing 8: Konkretes Beispiel

Die Werte in eine CSV-Datei zu speichern wurde deshalb gewählt, weil die Datensätze mit hoher über die bereits erläuterte Datenbankanweisung in die Datenbank geladen werden können.
Die gespeicherten Werte können auch über eine in PHP spezifizierte SQL-Anfrage direkt in die Datenbank geladen werden. Auch dieser Ansatz wurde bereits zu Beginn des Kapitels beschrieben und kann für die Integration über den XPath-Ansatz genutzt werden. Hierzu werden die nach dem XPath-Ausdruck gespeicherten Werte direkt in die Attributzuordnung der SQL-Anfrage eingetragen. Ein Nachteil liegt darin, dass jedes mal eine Reihe an SQL-Anfrage ausgeführt werden muss. Ein effektivere Lösung kann über eine *LOAD DATA LOCAL INFILE* Datenbankanweisung erreicht werden.
Der Ablauf für die Umsetzung der Datenintegration wird in Abschnitt 6.2 erläutert.

6.1.4. Vergleich der Implementierungsansätze

In diesem Abschnitt wird ein Vergleich der vorgestellten Implementierungsansätze vorgenommen. Hierzu werden in Tabelle 9 die drei Ansätze anhand unterschiedlicher Eigenschaften gegenübergestellt.

Eigenschaften	**Implementierungsansatz**		
	CSV	**Tool**	**PHP**
Flexibilität	-	±	+
Geschwindigkeit	+	+	±
Datenverarbeitung	+	+	+
Zugang zur DB	+	+	+
Benutzerfreundlichkeit	-	+	-
Betriebssystem	+	-	+

Tabelle 9: Zusammenfassender Vergleich der Implementierungsarten

PHP beinhaltet eine Vielzahl von Bibliotheken, mit denen verschiedene Funktionen umgesetzt werden können. PHP bietet den großen Vorteil, XPath-Ausdrücke anwenden zu können und somit Teile eines XML-Dokumentes zu adressieren. Die Adressierung über XPath kann weder mit dem Tool *MySQL for Excel*, noch über das CSV-Dateiformat realisiert werden.
PHP ist im Gegensatz zur Realisierung der Integration mittels des CSV-Dateiformates sowie dem Tool wesentlich flexibler, denn mittels PHP können sowohl einzelne Werte ausgewählt und zu einem Tupel zusammengefasst werden, als auch eine Menge von Werten direkt adressiert werden. Währenddessen mit dem CSV-Dateiformat lediglich gesamte Spalten ausgewählt werden. Über das Tool können zwar einzelne Werte ausgewählt und importiert werden, das Problem jedoch ist, dass diese Werte nicht zu einem Tupel zusammengefasst werden können. Aus diesem Grund stellt die Umsetzung über PHP den flexibelsten Lösungsansatz dar und wird daher für die Umsetzung der Datenintegration verwendet werden.
Bezüglich des Zugriffes auf die MySQL Datenbank ist kein Unterschied zwischen den Implementierungsansätzen festzustellen.
Das Tool bietet im Gegensatz zu PHP und dem CSV-Dateiformat die einfachste Anwendungsmöglichkeit für die Integration der Daten. Hierzu müssen lediglich die Datensätze der Eingabeprotokolle sowie die Tabelle der Datenbank ausgewählt werden. Der Import der Daten erfolgt anschließend automatisch. Die Umsetzung der Integration über PHP kann ausschließlich von einem Anwender mit dem notwendigen Hintergrundwissen implementiert werden. Auch die Integration der Datensätze über das CSV-Dateiformat muss durch eine Datenbankanweisung spezifiziert werden, die ebenfalls von einem Anwender mit dem notwendigen Hintergrundwissen implementiert werden muss. Das Tool kann mit einer geringen Einarbeitungszeit von jedem Anwender eingesetzt werden.
Bezugnehmend auf die Datenverarbeitung ist zwischen den Implementierungsansätzen kein Unterschied festzustellen. Die Daten werden problemlos aus den Excel-Eingabeprotokollen in die Datenbank transformiert.
Auch bezugnehmend auf die Geschwindigkeit ist kein großer Unterschied zu erkennen. Die Integration erfolgt sowohl über das CSV-Dateiformat und der anschließenden Datenban-

kanweisung, als auch über das Tool mit einer hohen Geschwindigkeit. Hinsichtlich PHP ist die Geschwindigkeit von der Anzahl der beinhalteten SQL *Insert Into* Anweisungen abhängig. Da jedoch die adressierten Werte aus Excel über die XPath-Ausdrücke direkt in eine CSV-Datei gespeichert werden, ist die Geschwindigkeit zu vernachlässigen, da die eigentliche Transformation der Daten über die *Load Data Local Infile*-Anweisung realisiert wird.
Das ausschlaggebende Merkmal für die Umsetzung der Adressierung aus den Eingabeprotokollen mit Hilfe von PHP ist zum einen auf die Flexibilität und zum anderen auf den Einsatz von XPath-Ausdrücken zurückzuführen.

6.2. Zusammenfassung der Datenintegration

In Abschnitt 6.1 wurden drei Implementierungsansätze vorgestellt. In diesem Abschnitt wird die Umsetzung des 3. Implementierungsansatzes zusammengefasst. Die Adressierung der zu integrierenden Daten erfolgt anhand von XPath-Ausdrücken. Die Schritte für die Umsetzung der Datenintegration sind im Folgenden aufgelistet.

1. Excel-Eingabeprotokolle in ein einheitliches Format (XML-Dateiformat) transformieren.
2. Über PHP-Skript, XPath-Ausdrücke generieren, mit deren Hilfe die Werte aus den eigentlichen Excel-Zellen adressiert werden.
3. Adressierte Werte in eine über PHP aufgebaute CSV-Datei ablegen.
4. Die in der CSV-Datei gespeicherten Werte anschließend über die Datenbankanweisung *LOAD DATA LOCAL INFILE* transformieren.
5. Werte in die Datenbank speichern.

Die aufgelisteten Schritte beschreiben den gesamten Ablauf der durchgeführten Datenintegration.
Die konkrete Transformationsanfrage wird über die Datenbankanweisung *LOAD DATA LOCAL INFILE* realisiert, mit deren Hilfe anschließend die Daten aus der CSV-Datei in die Datenbank geladen werden. Diese Anweisung ist in Listing 9 aufgelistet und wurde bereits detailliert in Abschnitt 6.1.1 beschrieben. Da die über XPath adressierten Werte in eine CSV-Datei gespeichert werden, kann diese Transformationsanfrage verwendet werden. Das Beispiel aus Listing 9 beschreibt die Transformation der adressierten Werte in die Datenbank-Tabelle *Station* aus Abbildung 52. Die Fremdschlüsselbeziehung wird ebenfalls über die Anweisung, mit Hilfe einer SQL-Abfrage, realisiert.
Hierzu muss der Name des Seegebietes im Excel-Eingabeprotokoll adressiert und ebenfalls in die CSV-Datei gespeichert werden. Der gespeicherte Wert muss anschließend über die *SELECT FROM WHERE* Klausel abgefragt und über *SET* in der Tabelle *Station* gespeichert werden.

```
1  LOAD DATA LOCAL INFILE 'station.csv'
2  INTO TABLE Station
3  fields terminated by ';'
4  enclosed by '"'
5  lines terminated by '\r' ignore 1 lines
6
7  (Longitude, Latitude, Name, @seegebietinit)
8
9     SET Seegebiet_idSeegebiet =
10    (SELECT idSeegebiet
11    FROM Seegebiet
12    WHERE @familieinit = Name);
```

Listing 9: **Load Data Local Infile** Anweisung

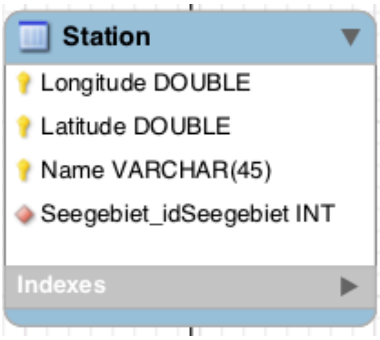


Abbildung 52: Tabelle "Station"

Über diese Art der Datenbankanweisung können sämtliche Werte in die Datenbank transformiert werden. Hierzu wurde die Adressierung in PHP umgesetzt. Anschließend wurden die adressierten Werte separat in eine CSV-Datei gespeichert. Die in der CSV-Datei gespeicherten Datensätze können zuletzt gemeinsam als Tupel in die Datenbank geladen werden. Dieser Implementierungsansatz stellt eine effektive Lösung für die Integration sämtlicher Daten aus den Excel-Eingabeprotokolle dar.

7. Zusammenfassung & Ausblick

In diesem Kapitel wird die vorliegende Arbeit zusammengefasst und anschließend ein Ausblick auf den weiteren Forschungsbedarf sowie Themen für mögliche Folgearbeiten gegeben.

Zu Beginn dieser Arbeit wurde in Kapitel 2 der biologische Hintergrund vorgestellt, wobei der Begriff *Benthos* definiert und die Gewinnung von Benthosproben der AG „Ökologie benthischer Organismen“ des IOW erläutert wurde. In diesem Zusammenhang wurden unter anderem Begriffe wie *Trockenmasse*, *Feuchtmasse* und *aschefreie Trockenmasse* geklärt, die grundlegend für die inhaltliche Bedeutung der in den Excel-Eingabeprotokollen gespeicherten Informationen sind.
In Kapitel 3 wurden die für die Arbeit benötigten Grundlagen vorgestellt, die in drei Abschnitte unterteilt sind. Im ersten Abschnitt 3.1 wurden Grundlagen bezüglich der Datenintegration vorgestellt, wobei zunächst die Grundprobleme wie *Verteilung*, *Autonomie* und *Heterogenität* der Datenintegration und anschließend sowohl die wesentlichen Integrationsschritte, als auch die Techniken zur Überwindung der definierten Grundprobleme, die sich hauptsächlich der Überwindung der Heterogenität zuwenden, beschrieben. Eine dieser Techniken ist das *Schema Mapping*, welches den Schwerpunkt für die Konzeption der Datenintegration darstellt. Im zweiten Abschnitt 3.2 wurden Grundlagen bezüglich der Datenbankkonzeption vorgestellt. Darunter zählen Aspekte wie der Datenbankentwurf, das ER-Modell, das relationale Datenbankmodell sowie das in der Arbeit verwendete DBMS MySQL. Der dritte Abschnitt befasst sich mit der Analyse von Excel-Tabelle. An dieser Stelle wurden zwei Ansätze vorgestellt, mit deren Hilfe eine umfassende Analyse von Tabellen ermöglicht werden kann. In diesem Kapitel wurden zusätzlich Themenfelder betrachtet, die nicht Bestandteil der vorliegenden Arbeit sind, jedoch für das Verständnis der Konzeption sowohl der Datenbank, als auch der Datenintegration grundlegend sind.
Um sowohl eine Datenintegration, als auch eine Datenbank konzipieren zu können, müssen die zur Verfügung stehenden Excel-Eingabeprotokolle analysiert werden. Hierzu wurde in Kapitel 4 eine konkrete Analyse der Eingabeprotokolle in Bezug auf Inhalt und Struktur vorgenommen. Aufgrund der historisch gewachsenen Struktur und den steigenden wissenschaftlichen Ansprüchen weisen die Eingabeprotokoll deutliche Unterschiede in der Struktur und dem Inhalt auf. Die Unterschiede führten dazu, dass sich zehn unterschiedliche Excel-Protokolltypen entwickelten. Die strukturellen und inhaltlichen Unterschiede wurden jeweils verdeutlicht. Im selben Kapitel wurden die auftretenden Probleme bezüglich der vorliegenden Excel-Eingabeprotokolle, hauptsächlich der Heterogenität betreffend, erläutert und Beispiele angebracht.
In Kapitel 5 wurde das Konzept der Arbeit vorgestellt. Im ersten Teil des Kapitels wurde das konkrete Datenbankkonzept beschrieben und dabei das für die Integration der Daten benötigte Zielschema modelliert. Das Zielschema wurde anhand des ER-Diagramms entworfen und wurde mit dem grafischen Benutzerwerkzeug *MySQL Workbench* durchgeführt. Der zweite Teil des Kapitels befasst sich mit der Thematik der Datenintegration. Hierzu wurde ein Konzept erstellt, mit deren Hilfe die Datenintegration der in Excel gespeicherten Daten durchgeführt werden kann. Das Integrationskonzept dieser Arbeit wurde auf Basis des Schema Mapping Prozesses entworfen. Zunächst wurden Wertkorrespondenzen zwischen

den lokalen Schemata, den Excel-Protokolltypen, und dem globalen Schema, das entworfene Zielschema, aufgestellt. Die Wertekorrespondenzen wurden anschließend interpretiert. Hierzu war es notwendig, die in den Excel-Eingabeprotokollen gespeicherten Werte zu lokalisieren und darauf aufbauend eine Adressierung der jeweiligen Excel-Zellen, in denen der zu integrierende Wert gespeichert ist, abzuleiten. Für die Adressierung der Werte wurden drei Ansätze vorgestellt. Der dritte Ansatz, der im weiteren Verlauf der Arbeit umgesetzt wurde, befasst sich mit der Adressierung über XPath-Ausdrücke, was gleichzeitig eine Konvertierung sämtlicher Excel-Eingabeprotokolle in das XML Format nach sich zieht. Das Problem hierbei stellen die zehn unterschiedlichen Protokolltypen dar, für die jeweils Muster definiert wurden. Die Muster beinhalten für jeden zu adressierenden Wert XPath-Ausdrücke, da in jedem der zehn Protokolltypen die Werte an unterschiedlichen Positionen zu lokalisieren sind. Im letzten Schritt des Integrationskonzeptes wurden die Transformationsanfragen vorgestellt, mit deren Hilfe die konkreten Daten der Excel-Eingabeprotokolle in die Struktur des Zielschemas überführt werden können.
In Kapitel 6 wurde die Implementierung des Datenintegrationskonzeptes beschrieben und die zuvor beschriebenen Aspekte des Konzeptes umgesetzt. Für die Umsetzung wurden drei Technologien verwendet, die jeweils für die Umsetzung der Datenintegration sowohl Vor-, als auch Nachteile aufweisen.

Die AG „Ökologie benthischer Organismen“ des IOW befasst sich mit der Untersuchung, Analyse und Bewertung benthischer Organismen. Die mit dieser Arbeit modellierte Datenbank bildet die Grundlagen für eine einheitliche und zentrale Speicherung bzw. Verwaltung der erfassten Daten. Das konzipierte Datenbankschema ist ebenso Grundlage für ein erweiterbares Datenbanksystem, welches unter anderem für weitere in diesem Zusammenhang durchzuführenden biologischen Anwendungen optimiert werden kann. Das modellierte Zielschema kann ohne weiteres um neue Anwendungsfelder bzw. Objekte erweitert werden und bildet somit eine Basis, um neue Daten in das Datenbanksystem zu speichern. Die ebenfalls konzipierte Datenintegration ist für die Integration der in Excel gespeicherten Daten zugrundeliegend. Mit den beschriebenen Lösungsansätzen können die Datensätze aus den Excel-Dokumenten lokalisiert, adressiert und anschließend in die Datenbank überführt werden. Das Erstellen eines Lösungsansatzes war notwendig, damit ein automatischer Import der Eingabeprotokolle erfolgen kann und somit keine manuelle Überführung der einzelnen Datensätze notwendig ist. Sämtliche in den Excel-Eingabeprotokollen gespeicherten Datensätze können mit Hilfe der erstellten XPath-Ausdrücke erfasst werden. Nicht nur das Erfassen der Daten aus den Excel-Dateien, sondern auch das Aufstellen von Transformationsanfragen ist notwendig, damit die Daten integriert werden können.
In dieser Arbeit wurden die Grundsteine für die gesamte Integration der Datenbestände aus den Excel-Eingabeprotokollen gelegt. Die Bausteine, die jeweils Teillösungen darstellen, müssen jeweils zu einer Gesamtlösung zusammengesetzt werden. Eine anknüpfende Folgearbeit sollte sich mit dieser Zusammenführung der Teillösungen und dem anschließenden automatischen Import befassen. Hierbei müssen insbesondere Fehler während der Integration betrachtet werden sowie gleichzeitig ein Data-Cleaning-Prozess, der in der vorliegenden Arbeit nicht konzipiert wurde, durchgeführt werden. Die Beseitigung von Fehler bezüglich

der Daten kann nicht immer automatisch erfolgen, sondern muss teilweise manuell durchgeführt werden.

Für die Adressierung der Daten wurden bereits Muster definiert, um jeweils die spätere Integration der Datenbestände eines bestimmten Protokolltyps zu vereinfachen. Die in diesen Mustern befindlichen XPath-Ausdrücke gewährleisten die Adressierung sämtlicher Daten der Protokolltypen und sind für eine Gesamtlösung zwingend notwendig. Das Auslesen der Daten aus jedem Protokolltyp wurde durch die Muster vervollständigt.

In Folgearbeiten sollte eine geeignete grafische Benutzeroberfläche erstellt werden, welche die Speicherung und Verwaltung der Datenbestände vereinfacht. Excel bot bisher eine einfache Anwendungsmöglichkeit und konnte mit wenig Einarbeitungszeit von jedem Anwender verwendet werden.

Des Weiteren ist es zwingend notwendig, die Integration der Datenbestände an einer Reihe von Excel-Eingabeprotokollen durchzuführen, um beispielsweise Datenausreißer erkennen zu können. Zusätzlich müssen Abweichungen von den in dieser Arbeit spezifizierten Mustern erkannt werden, da durchaus der Fall eintreten kann, dass die zu integrierenden Werte nicht an den vorgegeben Positionen gespeichert sind, sondern eventuell falsch positioniert wurden. Diese möglichen Fehler sollten durch einen geeigneten Lösungsansatz überwunden werden.

Bisher wurden die Transformationsanfragen mit Hilfe von SQL umgesetzt. Eine weitere Möglichkeit ist die Verwendung der Abfragesprache XQuery. XQuery benutzt unter anderem XPath und bietet eine Möglichkeit die Daten direkt zusammenzuführen.

Wie bereits zu Beginn der Arbeit angesprochen, soll ein automatischer Taxonomiecheck der in der Datenbank gespeicherten Taxa erfolgen. Dieser Aspekt könnte ebenfalls in Zusammenhang mit weiteren Arbeiten auf dem Gebiet betrachtet werden. Hierzu muss eine Anbindung der Datenbank an das *World Register of Marine Species* (WoRMS) realisiert werden.

Literaturverzeichnis

[1] : MySQL Referenzhandbuch :: 13.2.5 LOAD DATA INFILE. https://dev.mysql.com/doc/refman/5.1/de/load-data.html, 2010, Abruf 13.04.2014.

[2] : Überlegenheit einer Datenbank im Vergleich zu einer Excel-Lösung. http://www.lorenzsoft.de/was-wir-tun/datenbank-im-vergleich-zu-einer-excel-loesung.html, Abruf 13.02.2014.

[3] : MySQL 5.5 Reference Manual :: 26 MySQL for Excel. http://dev.mysql.com/doc/refman/5.5/en/mysql-for-excel.html, Abruf 13.04.2014.

[4] : PHP-Handbuch :: Einführung :: Was ist PHP. http://www.php.net/manual/de/intro-whatis.php, Abruf 14.04.2014.

[5] : PHP-Handbuch. http://www.php.net/manual/de/, Abruf 16.04.2014.

[6] : Das Leibniz-Institut für Ostseeforschung Warnemünde. http://www.io-warnemuende.de, Abruf 20.01.2014.

[7] Baker M. : PHPExcel - OpenXML - Read, Write and Create Excel documents in PHP - Spreadsheet engine, Version 127. https://phpexcel.codeplex.com, März 2014, Abruf 03.04.2014.

[8] Bruder I., Heuer A. : Informationsintegration. Vorlesung, Uni Rostock, 2012/13.

[9] Cox M. : Schlüssel. http://www.ibexpert.net/ibe_de/index.php?n=Doku.Schluessel, Mai 2013, Abruf 20.03.2014.

[10] Darr A., Schiele K., Glück F., Zettler M. : Bodenlebende Wirbellose als Indikator in der Umweltüberwachung. *Leibniz-Institut für Ostseeforschung Warnemünde*, Poster, Erhalten: 09.2013 vom IOW.

[11] Doush I.,Pontelli E. : Detecting and Recognizing Tables in Spreadsheets. Technical report, International Journal on Document Analysis and Recognition, 2010.

[12] Feistel R., Nausch G., Wasmund N. *: State and evolution of the Baltic Sea, 1952 – 2005, a detailed 50-year survey of meteorology and climate, physics, chemistry, biology, and marine environment., Chapter 17: Zoobenthos, Zettler et al.* Hoboken: Wiley-Interscience, 2008.

[13] Forster S. : Die unsichtbaren Bewohner des Meeresbodens. http://www.io-warnemuende.de/die-unsichtbaren-bewohner-des-meeresbodens.html, August 2006, Abruf 20.01.2014.

[14] Glatz K. : Software zur automatischen Zuordnung von ähnlichen Datenbank-Schemata inklusive Datenmigration und Testdatenerzeugung. Bachelorarbeit, Hochschule Ravensburg-Weingarten, März 2009.

[15] Glück F., Darr A., Schiele K. : Was ist Benthos. *Leibniz-Institut für Ostseeforschung Warnemünde*, Poster, Erhalten: 09.2013 vom IOW.

[16] Glück F., Darr A., Zettler M. : Was lebt am Grund der Ostsee. *Leibniz-Institut für Ostseeforschung Warnemünde*, Poster, Erhalten: 09.2013 vom IOW.

[17] Glück F., Keiser N., Darr A., Pohl F. : Die Gewinnung von Benthosproben. *Leibniz-Institut fuer Ostseeforschung Warnemuende*, Poster, Erhalten: 09.2013 vom IOW.

[18] Haas L. : Beauty and the Beast: The Theory and Practice of Information Integration. In *ICDT'07 Proceedings of the 11th international conference on Database Theory*, pages 28–43. Springer Verlag Berlin, Heidelberg, 2007.

[19] Herschel M. : Datenintegration und Datenherkunft. Vorlesung, Lehrstuhl für Datenbanksysteme, Uni Tuebingen, 2010/11.

[20] IT Wissen, das große Online-Lexikon für Informationstechnologie. : XPath (XML Path Language). `http://www.itwissen.info/definition/lexikon/XPath-XML-Path-Language.html`, Abruf 25.04.2014.

[21] Kemper A., Eickler A. *: Datenbanksysteme.* Oldenbourg, 6. Auflage, 2006.

[22] Leibniz-Institut für Ostseeforschung Warnemünde. : Prüfanweisung MZB AMS, Auswertung von Makrozoobenthos - Proben aus marinen Sedimenten. Leibniz-Institut für Ostseeforschung Warnemünde, Februar 2014.

[23] Leser U. : Informationsintegration: Verteilung, Autonomie. Vorlesung, Humboldt Universität zu Berlin, 2008/09.

[24] Leser U., Naumann F. *: Informationsintegration, Architektur und Methoden zur Integration verteilter und heterogener Datenquellen.* dpunkt.verlag, 1. Auflage, 2007.

[25] Naumann F. : Informationsintegration: Schema Mapping. Vorlesung, Humboldt Universität zu Berlin, 2006.

[26] Pröll S., Zangerle E., Gassler W. *: MySQL, Das umfassende Handbuch.* Galileo Computing, 2. Auflage, 2013.

[27] Rheinheimer G. *: Meereskunde der Ostsee.* Springer, 2. Auflage, 1996.

[28] Rossak I. *: Datenintegration.* HANSER Verlag, 1. Auflage, 2013.

[29] Sattler K., Saake G., Heuer A. *: Datenbanken, Konzepte und Sprachen.* mitp, 4. Auflage, 2010.

[30] Schnabel J. : Formen der Heterogenität. Vorlesung, TU Kaiserslautern, 2004.

[31] Schuldt H. : Kapitel 3: Informationsintegration. Vorlesung, Uni Basel, 2014.

[32] Straube G. : Datenintegration für die global-as-local-view-extension-Technik. Master's thesis, Uni Rostock, 2013.

[33] Thor A. : Datenintegration; Eigenschaften von Integrationssystemen. Technical report, Uni Leipzig, 2008.

[34] Thor A. : Datenintegration; Kapitel 6: Schemamanagement. Vorlesung, Uni Leipzig, 2008.

[35] VIO.Matrix Support. : CSV Dateien dynamisch erzeugen. `http://www.viomatrix.de/programmierung-csv-dateien-dynamisch-erzeugen.html`, Abruf 14.04.14.

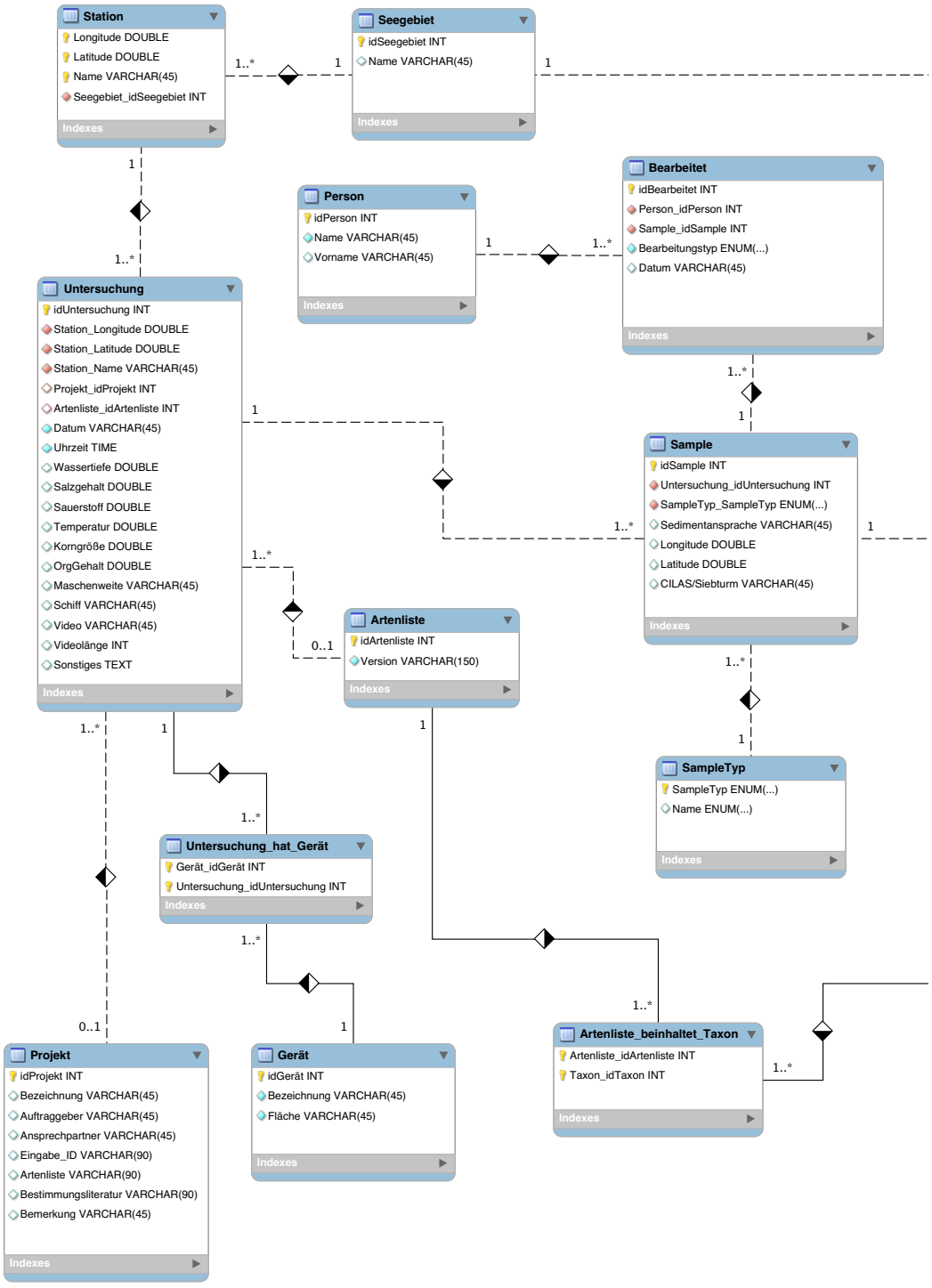
[36] Wang X. : Tabular Abstraction, Editing and Formatting. University of Waterloo, 1996.

[37] Zanibbi R., Blostein D., Cordy J. : A survey of table recognition. In *International Journal on Document Analysis and Recognition*, pages 1–16. Springer Verlag, 2004.

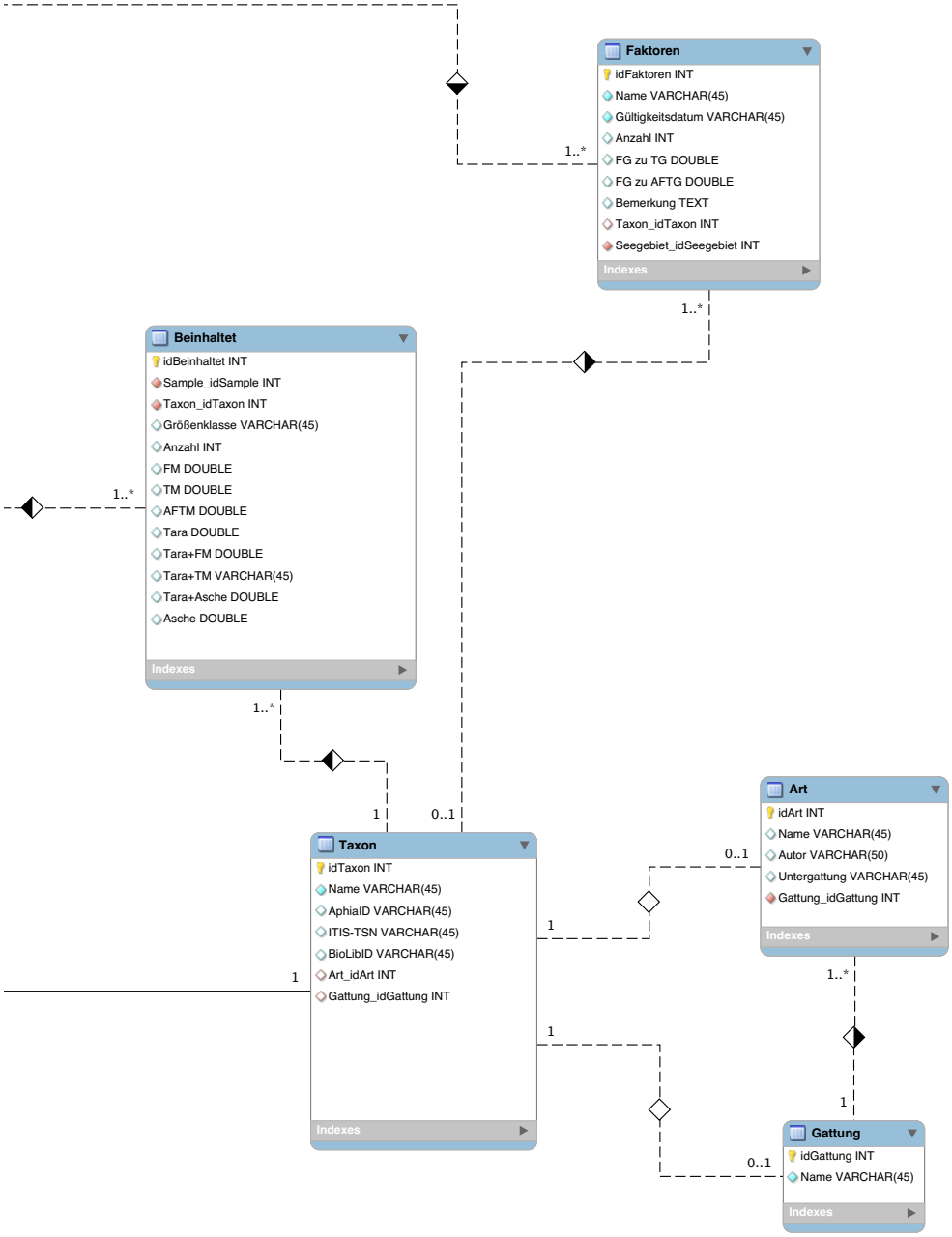
A. Anhang zum Konzept

A.1. Erweitertes ER-Diagramm der Benthos-Datenbank

Seite 1 von 2



Seite 2 von 2


Faktoren
idFaktoren INT
Name VARCHAR(45)
Gültigkeitsdatum VARCHAR(45)
Anzahl INT
FG zu TG DOUBLE
FG zu AFTG DOUBLE
Bemerkung TEXT
Taxon_idTaxon INT
Seegebiet_idSeegebiet INT
Indexes
1..*
Beinhaltet
idBeinhaltet INT
Sample_idSample INT
Taxon_idTaxon INT
Größenklasse VARCHAR(45)
Anzahl INT
FM DOUBLE
TM DOUBLE
AFTM DOUBLE
Tara DOUBLE
Tara+FM DOUBLE
Tara+TM VARCHAR(45)
Tara+Asche DOUBLE
Asche DOUBLE
Indexes
1..*
1
0..1
Taxon
idTaxon INT
Name VARCHAR(45)
AphiaID VARCHAR(45)
ITIS-TSN VARCHAR(45)
BioLibID VARCHAR(45)
Art_idArt INT
Gattung_idGattung INT
Indexes
Art
idArt INT
Name VARCHAR(45)
Autor VARCHAR(50)
Untergattung VARCHAR(45)
Gattung_idGattung INT
Indexes
Gattung
idGattung INT
Name VARCHAR(45)
Indexes

A.2. Definierte Muster Navigationsrichtung

Excel-Protokolltyp Nr. 10

Navigationsrichtungen für die Lokalisierung der Werte aus dem Quellschema in das Zielschema unter Angabe der Bezeichnung der jeweiligen Datenbanktabelle.

Datenbanktabelle *Untersuchung*

Attributbezeichnung Zielschema	Attributbezeichnung Quellschema	Navigation zum Wert
Datum	Datum Probenahme:	Spalte + a
Uhrzeit	Zeit (lokal)	Spalte + a
Wassertiefe	Wassertiefe (m)	1. Spalte + a 2. Zeile + b
Sauerstoff	Sauerstoff (mg/l)	Spalte + a
Temperatur	Temperatur (°C)	Spalte + a
Korngröße	d50	Spalte + a
OrgGehalt	Org. (%)	Spalte + a
Maschenweite	Maschenweite:	Spalte + a
Schiff	Schiff	Spalte + a
Video	Video (Anlage&Dauer)	-
Videolänge	Video (Anlage&Dauer)	
Sonstiges	Sonstige	Spalte + a

Datenbanktabelle *Seegebiet* und *Artenliste*

Attributbezeichnung Zielschema	Attributbezeichnung Quellschema	Navigation zum Wert
Name	Seegebiet:	Spalte + a

Attributbezeichnung Zielschema	Attributbezeichnung Quellschema	Navigation zum Wert
Version	Gültige Artenliste:	Spalte + a

Datenbanktabelle *Sample* und *SampleTyp*

Attributbezeichnung Zielschema	Attributbezeichnung Quellschema	Navigation zum Wert
Sedimentansprache	Sedimentansprache Bord	Spalte + a
Longitude	Position E°min,dec	Spalte + a
Latitude	Position N°min,dec	Spalte + a
CILAS/Siebturm	CILAS/Siebturm	Zeile + b

Attributbezeichnung Zielschema	Attributbezeichnung Quellschema	Navigation zum Wert
Name	Hol	Spalte + a

Datenbanktabelle *Beinhaltet* und *Projekt*

Attributbezeichnung Zielschema	Attributbezeichnung Quellschema	Navigation zum Wert
Größenklasse	Größenklasse (mm)	Zeile + b
Anzahl	Anzahl Individuen	Zeile + b
FM	WW (g per sample)	Zeile + b
TM	DW (g per sample)	Zeile + b
AFTM	AFDW (g per sample)	Zeile + b
Tara	Tara	Zeile + b
Tara+FM	Tara + FM (g)	Zeile + b
Tara+TM	Tara + TM (g)	Zeile + b
Tara+Asche	Tara + asche (g)	Zeile + b
Asche	Asche (g)	Zeile + b

Attributbezeichnung Zielschema	Attributbezeichnung Quellschema	Navigation zum Wert
Bezeichnung	Projektbezeichnung:	Spalte + a
Auftraggeber	Auftraggeber:	Spalte + a
Ansprechpartner	Ansprechpartner:	1. Spalte + a 2. Zeile + b
Eingabe_ID	ID:	Spalte + a
Artenliste	gültige Artenliste:	Spalte + a
Bestimmungsliteratur	gültige Liste Bestimmungsliteratur:	Spalte + a
Bemerkung	-	-

Datenbanktabelle *Station* und *Person*

Attributbezeichnung Zielschema	Attributbezeichnung Quellschema	Navigation zum Wert
Longitude	E°min,dec	Spalte + a
Latitude	N°min,dec	Spalte + a
Name	Stationsname:	Spalte + a

Attributbezeichnung Zielschema	Attributbezeichnung Quellschema	Navigation zum Wert
Name	Bearbeiter_Labor, Wäger_FM, Wäger_TM, Wäger_AFTM, QK Eingabe Bearbeiter Korngröße, Bearbeiter Organik	Spalte + a Zeile + b
Vorname		

Datenbanktabelle *Taxon* und *Art*

Attributbezeichnung Zielschema	Attributbezeichnung Quellschema	Navigation zum Wert
Name	valider Artname_ohne_Autor_Jahr	Zeile + b
AphiaID	AphiaID	Zeile + b
ITIS-TSN	ITIS-TSN	Zeile +b
BioLibID		

Attributbezeichnung Zielschema	Attributbezeichnung Quellschema	Navigation zum Wert
Name	Art	Zeile + b
Autor	valider Artname_Autor_Jahr	Zeile + b
Untergattung		

Datenbanktabelle *Bearbeitet* und *Faktoren*

Attributbezeichnung Zielschema	Attributbezeichnung Quellschema	Navigation zum Wert
Bearbeitungstyp	Bearbeiter_Labor, Wäger_FM, Wäger_TM, Wäger_AFTM, QK Eingabe Bearbeiter Korngröße, Bearbeiter Organik	
Datum	Datum Laborbearbeitung, Datum Wägung_FM, Datum Wägung_TM, Datum Wägung_AFTM, QK Eingabe	

Attributbezeichnung Zielschema	Attributbezeichnung Quellschema	Navigation zum Wert
Name	Art	Zeile + b
Gültigkeitsdatum		
Anzahl	Anzahl	Zeile + b
FGzuTG	FG→TG	Zeile + b
FGzuAFTG	FG→AFTG	Zeile + b
Bemerkung		

Datenbanktabelle *Gattung* und *Gerät*

Attributbezeichnung Zielschema	Attributbezeichnung Quellschema	Navigation zum Wert
Name	Gattung	Zeile + b

Attributbezeichnung Zielschema	Attributbezeichnung Quellschema	Navigation zum Wert
Bezeichnung	Gerät	Spalte + a
Fläche	Fläche/Hol	Spalte + a

A.3. Definierte Muster XPath-Ausdrücke

Die nachfolgenden Muster zeigen jeweils einen Auschnitt der angegeben Excel-Protokolltypen. Die kompletten Muster für die Excel-Protokolltypen 1, 2 und 10 sind auf der beigefügten DVD zu finden.

Excel-Protokolltyp Nr. 1 (Seite 1/1)

EXCEL - PROTOKOLLTYP NR. 1

Tabelle „Projekt“
keine Daten vorhanden!

idProjekt:
auto increment (wird nicht generiert)

Bezeichnung:	nicht vorhanden
Auftraggeber:	nicht vorhanden
Ansprechpartner:	nicht vorhanden
Eingabe_ID:	nicht vorhanden
Artenliste:	nicht vorhanden
Bestimmungsliteratur:	nicht vorhanden
Bemerkung:	nicht vorhanden

Tabelle „Seegebiet“

idSeegebiet:
auto increment (wird nicht generiert)

Name: nicht vorhanden, muss manuell eingetragen werden

Tabelle „Station“

Longitude:
//ss:Data[text() = 'Koordinaten:']/parent::*/parent::*/following-sibling::*[1]/ss:Cell/following-sibling::*[1]/ss:Data

Latitude:
//ss:Data[text() = 'Koordinaten:']/parent::*/following-sibling::*[1]/ss:Data

Name:
//ss:Data[text() = 'Station:']/parent::*/following-sibling::*[1]/ss:Data

Seegebiet_idSeegebiet:
Schlüssel aus Tabelle Seegebiet (idSeegebiet)

Tabelle „Artenliste“
keine Daten vorhanden!

idArtenliste:
auto increment (wird nicht generiert)

Name: nicht vorhanden

Excel-Protokolltyp Nr. 2 (Seite 1/2)

EXCEL - PROTOKOLLTYP NR. 2

Tabelle „Projekt“
keine Daten vorhanden!

idProjekt:
auto increment (wird nicht generiert)

Bezeichnung:	nicht vorhanden
Auftraggeber:	nicht vorhanden
Ansprechpartner:	nicht vorhanden
Eingabe_ID:	nicht vorhanden
Artenliste:	nicht vorhanden
Bestimmungsliteratur:	nicht vorhanden
Bemerkung:	nicht vorhanden

Tabelle „Seegebiet“
keine Daten vorhanden!

idSeegebiet:
auto increment (wird nicht generiert)

Name:	nicht vorhanden, muss manuell eingetragen werden

Excel-Protokolltyp Nr. 2 (Seite 2/2)

Tabelle „Station“

Longitude:
//ss:Row[5]/ss:Cell/ss:Data[text() = 'Koordinaten:']/parent::*/parent::*/parent::*/ss:Row[6]/ss:Cell[9]/ss:Data

Latitude:
//ss:Row[5]/ss:Cell/ss:Data[text() = 'Koordinaten:']/parent::*/following-sibling::*[2]/ss:Data

Name:
//ss:Row[3]/ss:Cell/ss:Data[text() = 'Station:']/parent::*/following-sibling::*[1]/ss:Data

Seegebiet_idSeegebiet:
Schlüssel aus Tabelle Seegebiet (idSeegebiet)

Tabelle „Artenliste“
keine Daten vorhanden!

idArtenliste:
auto increment (wird nicht generiert)

Name: nicht vorhanden
Version: nicht vorhanden

Tabelle „Gerät“

idGerät:
auto increment

Name:
//ss:Row[3]/ss:Cell/ss:Data[text() = ‘Gerät:']/parent::*/following-sibling::*[2]/ss:Data

Fläche:
//ss:Row[5]/ss:Cell/ss:Data[text() = 'Fläche cm² je']/parent::*/following-sibling::*[2]/ss:Data

Tabelle „Untersuchung_hat_Gerät“

Untersuchung_idUntersuchung:
Schlüssel aus Untersuchung

Gerät_idGerät:
Schlüssel aus Gerät

Tabelle „Untersuchung“

idUntersuchung: auto increment
Station_Longitude: Schlüssel aus Station

Excel-Protokolltyp Nr. 10 (Seite 1/3)

EXCEL - PROTOKOLLTYP NR. 10

Tabelle „Projekt“

idProjekt:
auto increment

Bezeichnung:
//ss:Worksheet[@ss:Name = 'EingabeMetadaten']/ss:Table/ss:Row/ss:Cell/ss:Data[text() = 'Projektbezeichnung:']/parent::*/following-sibling::*[2]/ss:Data

Auftraggeber:
//ss:Worksheet[@ss:Name = 'EingabeMetadaten']/ss:Table/ss:Row/ss:Cell/ss:Data[text() = 'Auftraggeber:']/parent::*/following-sibling::*[2]/ss:Data

Ansprechpartner:
//ss:Worksheet[@ss:Name = 'EingabeMetadaten']/ss:Table/ss:Row/ss:Cell/ss:Data[text() = 'Ansprechpartner:']/parent::*/parent::*/following-sibling::*[1]/ss:Cell/following-sibling::*[6]/ss:Data

Eingabe_ID:
//ss:Worksheet[@ss:Name = 'EingabeMetadaten']/ss:Table/ss:Row/ss:Cell/ss:Data[text() = 'ID:']/parent::*/following-sibling::*[1]/ss:Data

Artenliste:
//ss:Worksheet[@ss:Name = 'EingabeMetadaten']/ss:Table/ss:Row/ss:Cell/ss:Data[text() = 'gültige Artenliste:']/parent::*/following-sibling::*[2]/ss:Data

Bestimmungsliteratur:
//ss:Worksheet[@ss:Name = 'EingabeMetadaten']/ss:Table/ss:Row/ss:Cell/ss:Data[text() = 'gültige Liste Bestimmungsliteratur:']/parent::*/following-sibling::*[2]/ss:Data

Bemerkung:
nicht vorhanden!

Tabelle „Seegebiet“

idSeegebiet:
auto increment

Name:
//ss:Worksheet[@ss:Name = 'EingabeMetadaten']/ss:Table/ss:Row/ss:Cell/ss:Data[text() = 'Seegebiet:']/parent::*/following-sibling::*[2]/ss:Data

Excel-Protokolltyp Nr. 10 (Seite 2/3)

Tabelle „Station“

Longitude:
//ss:Worksheet[@ss:Name = 'EingabeMetadaten']/ss:Table/ss:Row/ss:Cell/ss:Data[text() = 'Positionierung (WGS 84):']/parent::*/parent::*/following-sibling::*[1]/ss:Cell/ss:Data[text() = 'E°min,dec']/parent::*/following-sibling::*[2]/ss:Data

Latitude:
//ss:Worksheet[@ss:Name = 'EingabeMetadaten']/ss:Table/ss:Row/ss:Cell/ss:Data[text() = 'Positionierung (WGS 84):']/parent::*/parent::*/following-sibling::*[2]/ss:Cell/ss:Data[text() = 'N°min,dec']/parent::*/following-sibling::*[2]/ss:Data

Name:
//ss:Worksheet[@ss:Name = 'EingabeMetadaten']/ss:Table/ss:Row/ss:Cell/ss:Data[text() = 'Stationsname:']/parent::*/following-sibling::*[1]/ss:Data

Seegebiet_idSeegebiet:
Schlüssel aus Tabelle Seegebiet (idSeegebiet)

Tabelle „Artenliste“

idArtenliste:
auto increment

Name:
//ss:Worksheet[12]/@*

Version:
//ss:Worksheet[@ss:Name = 'EingabeMetadaten']/ss:Table/ss:Row/ss:Cell/ss:Data[text() = 'gültige Artenliste:']/parent::*/following-sibling::*[2]/ss:Data

Tabelle „Gerät“

idGerät:
auto increment

Name:
//ss:Worksheet[@ss:Name = 'EingabeMetadaten']/ss:Table/ss:Row/ss:Cell/ss:Data[text() = 'Gerät']/parent::*/following-sibling::*/ss:Data **-> 1 bis 4**

Fläche:
//ss:Worksheet[@ss:Name = 'EingabeMetadaten']/ss:Table/ss:Row/ss:Cell/ss:Data[text() = 'Fläche/ Hol']/parent::*/following-sibling::*/ss:Data **-> 1 bis 4**

Excel-Protokolltyp Nr. 10 (Seite 3/3)

//ss:Worksheet[@ss:Name = 'Eingabe_Hol1']/ss:Table/ss:Row/ss:Cell/ss:Data[text() = 'Tara + asche (g)']/parent::*/parent::*/following-sibling::ss:Row/ss:Cell[1]/following-sibling::*[9]/ss:Data
Asche:
//ss:Worksheet[@ss:Name = 'Eingabe_Hol1']/ss:Table/ss:Row/ss:Cell/ss:Data[text() = 'Asche (g)']/parent::*/parent::*/following-sibling::ss:Row/ss:Cell[1]/following-sibling::*[10]/ss:Data

Tabelle „Person“
22 Zellen müssen adressiert werden!

idPerson: auto increment

Name: 22mal
Hol 1,2,3 & D:
//ss:Data[text() = 'Hol']/ancestor::*//ss:Data[text() = ‘1 bis D’]/ancestor::*/following-sibling::*/ss:Cell/ss:Data[text() = 'Bearbeiter Labor & Wäger_FM & Wäger_TM & Wäger_AFTM & Qk Eingabe']/parent::*/following-sibling::*[1]/ss:Data
Sediment:
//ss:Data[text() = 'Sediment']/parent::*/parent::*/following-sibling::*/ss:Cell/ss:Data[text() = 'Bearbeiter Organik:']/parent::*/parent::*/following-sibling::*[1]/ss:Cell[6]/ss:Data

//ss:Data[text() = 'Sediment']/parent::*/parent::*/following-sibling::*/ss:Cell/ss:Data[text() = 'Bearbeiter Korngröße:']/parent::*/parent::*/following-sibling::*[1]/ss:Cell[6]/ss:Data

Vorname:
nicht in Excel Datei explizit zu adressieren; muss über php vorgenommen werden

alle Namen aus einer Zeile (jew. mit 5 zeilen durchführen; gibt Menge zurück (//ss:Worksheet[@ss:Name = 'EingabeMetadaten']/ss:Table/ss:Row[26]/ss:Cell[position() >1]/ss:Data)

Tabelle „Bearbeitet“

idBearbeitet:	auto increment
Person_idPerson:	Fremdschlüssel aus Tabelle Person
Sample_idSample:	Fremdschlüssel aus Tabelle Sample
Sample_Untersuchung_idUntersuchung:	Fremdschlüssel aus Tabelle Sample
Bearbeitungstyp:	siehe unten
Datum:	siehe unten

Bearbeiter Labor, Hol 1 bis D =
//ss:Data[text() = 'Hol']/ancestor::*//ss:Data[text() = ‘1 bis D’]/ancestor::*/following-sibling::*/ss:Cell/ss:Data[text() = 'Bearbeiter Labor']/parent::*/following-sibling::*[1 bis 4]/ss:Data

Datum Laborbearbeitung, Hol 1 bis D =
//ss:Data[text() = 'Hol']/ancestor::*//ss:Data[text() = ‘1 bis D’]/ancestor::*/following-sibling::*/ss:Cell/ss:Data[text() = 'Datum Laborbearbeitung']/parent::*/following-sibling::*[1 bis D]/ss:Data

B. Anhang zur Implementierung

B.1. 1. Möglichkeit

```
1  // Verbindung zur DB herstellen
2  mysql_connect('localhost', 'username', 'password');
3
4  // Name der DB angeben
5  mysql_select_db('BenthosDB');
6
7  // PHPExcel-Bibliothek einbinden
8  require_once 'reader.php';
9
10 // neues Excel Objekt generieren; aus
11 // Bibliothek
12 $objExcel = new Spreadsheet_Excel_Reader();
13
14 // Name der Excel-Datei angeben --> in .xls Format bringen
15 $objExcel->read('....xls');
16
17 $i = $objExcel->sheets[0]['numRows'];
18
19 // Start
20 // Tabelle SEEGEBIET füllen
21 // SQL Anfrage generieren
22 // SQL INSERT Anweisung; Tabelle SEEGEBIET füllen
23     $strQuery = "INSERT INTO `Seegebiet` SET
24     `Name` = '" . $objExcel->sheets[0]['cells'][8][7] . "'";
25      // idSeegebiet --> Schlüssel (AI) und
26      // muss nicht mit angegeben werden
27
28
29
30 // Tabelle STATION füllen, Fremdschlüssel beachten
31 // 1.   Anfrage nach Fremdschlüssel `Seegebiet_idSeegebiet
32 $query = "SELECT idSeegebiet
33                 FROM Seegebiet
34                 WHERE Name = '" . $objExcel->
35                                   sheets[0]['cells'][8][7] . "'";
36 $result = mysql_query($query);
37 $row = mysql_fetch_assoc($result);
38
39 // 2.   SQL Anfrage generieren
40 // SQL INSERT Anweisung; Tabelle STATION füllen
41     $strQuery = "INSERT INTO `Station` SET
42     `Longitude` = '" . $objExcel->sheets[0]['cells'][11][7] . "',
43     `Latitude` = '" . $objExcel->sheets[0]['cells'][12][7] . "',
44     `Name` = '" . $objExcel->sheets[0]['cells'][8][2] . "',
45     // auf Fremdschlüssel referenzieren
46     // Ergebnis aus Anfrage nutzen
47     `Seegebiet_idSeegebiet` = '" . $row['idSeegebiet'] . "'";
```

Listing 10: Konkretes Beispiel für die Adressierung über PHPExcel-Bibliothek

Eidesstattliche Erklärung

Eidesstattliche Erklärung zur Masterarbeit

Ich, Jessica Zierke, versichere, die von mir vorgelegte Arbeit zu dem Thema "Konzeption der Datenintegration für eine zu entwickelnde Benthos-Datenbank" selbstständig verfasst zu haben. Alle Stellen, die wörtlich oder sinngemäß aus veröffentlichten oder nicht veröffentlichten Arbeiten anderer entnommen sind, habe ich als entnommen kenntlich gemacht. Sämtliche Quellen und Hilfsmittel, die ich für die Arbeit benutzt habe, sind angegeben. Die Arbeit hat mit gleichem Inhalt bzw. in wesentlichen Teilen noch keiner anderen Prüfungsbehörde vorgelegen.

$Unterschrift:$ $Ort, Datum:$