Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences

DEPARTMENT INFORMATION

Bachelorarbeit

Inslex - Ein webbasiertes Wörterbuch Hochdeutsch-Niederdeutsch für das Institut für Niederdeutsche Sprache (INS) Bremen. Konzeption und Umsetzung von Interface und Programmlogik in einem Content Management System

vorgelegt von
Felix Vollmuth

Studiengang Bibliotheks- und Informationsmanagement

erster Prüfer: Prof. Dr. Franziskus Geeb
zweiter Prüfer: Prof. Dr. Martin Gennis

Hamburg, April 2011

Abstract

Die Arbeit beschäftigt sich mit dem Entwurf und der Umsetzung eine Onlinewörterbuchs Hochdeutsch – Niederdeutsch (*inslex*) für das Institut für niederdeutsche Sprache. Ausgehend von den lexikographischen Grundlagen wird eine Wörterbuchstruktur erarbeitet. Diese wird als MySQL Datenbank umgesetzt und über eine Weboberfläche zugänglich gemacht. Es wird eine Beschreibung des fertigen Systems erstellt, sowie der Programmcode erläutert.

Schlagworte

Datenbank, Institut für niederdeutsche Sprache, jQuery, Lexikographie, MySQL, Niederdeutsch, PHP, Plattdeutsch, Wörterbuch

Inhaltsverzeichnis

Anhangsverzeichnis

Abbildungsverzeichnis

Einleitung

Das Institut für niederdeutsche Sprache Bremen (INS) betreibt auf seiner Webseite ein Onlinewörterbuch für Hochdeutsch-Plattdeutsch Übersetzungen mit dem Namen *inslex*. In diesem Wörterbuch werden Übersetzungen von Wörtern gesammelt, welche so in Nachrichtenmeldungen in plattdeutscher Sprache des Norddeutschen Rundfunks (NDR) verwendet wurden. Im Laufe der Zeit haben sich die Anforderungen an *inslex* gewandelt, so dass von Seiten des INS eine Neuentwicklung beschlossen wurde.

Ziel dieser Arbeit ist es die zugrunde liegende Technik für *inslex* zu entwickeln. Ausgehend vom bereits bestehenden Datenbestand und den Anforderungen des INS wird die Struktur des Wörterbuchs geplant. Diese Struktur wird sodann als MySQL Datenbank umgesetzt. Über ein Webformular kann mittels PHP-Scripte auf die Datenbank zugegriffen werden. Die Benutzerschnittstelle wird mithilfe von JavaScript mit weiteren Funktionen versehen, hierbei kommt die JavaScript-Bibliothek jQuery zum Einsatz. Der Schwerpunkt liegt in der praktischen Umsetzung, der vorliegende Text soll diese dokumentieren und erläutern.

Der Text gliedert sich in drei Teile: Zuerst wird eine allgemeine, theoretische Grundlage geschaffen, sowie die Anforderungen an das Wörterbuch ermittelt. Davon ausgehend wird eine Struktur für die Neuauflage von *inslex* erarbeitet. Der zweite Teil beschreibt diese neue Version und gibt Hinweise zur Bedienung. Der dritte Teil beschäftigt sich mit der Datenbank von einem technischen Standpunkt aus und gibt einen Einblick wie bestimmte Funktionen der Benutzerschnittstellen programmiertechnisch gestaltet wurden.

Der Fokus liegt auf den technischen Aspekten der Erstellung eines Onlinewörterbuches; allgemeine lexikographische Grundlagen werden nur insofern behandelt, wie sie zur Entwicklung von *inslex* und dem Verständnis des fertigen Systems beitragen. Die Datenbasis für das Wörterbuch besteht bereits, so dass Fragen der Lemmaauswahl oder der Übersetzung nicht angesprochen werden. Die Eingabe der Daten in das Produktivsystem ist nicht mehr Bestandteil dieser Arbeit. Die zentrale Frage lautet: Wie kann aus der vorliegenden Datenbasis eine Struktur für ein Wörterbuch entwickelt werden und diese in einer Datenbank umgesetzt werden?

1 Allgemein-theoretischer Teil

1.1 Lexikographie

1.1.1 Definition Wörterbuch

Im Folgenden soll das *„Wörterbuch“*, wie es im weiteren Verlauf benutzt wird, definiert werden. Die Wissenschaft der Erstellung von (Sprach)-wörterbüchern ist die Lexikographie (vgl. Schlaefer 2002, S.74). Sie ein Teilgebiet der Lexikologie, welche sich mit dem Wortschatz und den Wörtern beschäftigt (vgl. Schlaefer 2002, S. 12). Die übergeordnete Wissenschaft ist die Lexikontheorie, welche den menschlichen Wortschatz an sich erforscht und die allgemeinen theoretischen Grundlagen bereitstellt (vgl. Carstensenet et al. 2004, S. 434).

Eine passende Definition für ein Wörterbuch im Sinne dieser Arbeit lautet:

„Das Wörterbuch ist eine durch ein bestimmtes Medium repräsentierte Sammlung, von lexikalischen Einheiten (vor allem Wörtern), zu denen für einen bestimmten Benutzer bestimmte Informationen gegeben werden, die so geordnet sein müssen, daß ein rascher Zugang zu Einzelinformationen möglich ist“
(Schwarze / Wunderlich 1985, S.369)

Diese Definition spricht gleich mehrere relevante Punkte an: Ein Wörterbuch ist eine *„Sammlung“*, es muss sich nicht zwangsläufig um ein Buch (aus Papier) handeln. Somit fallen z.B. auch elektronische Werke unter diese Definition.

Die Informationen müssen so geordnet sein, dass *„ein rascher Zugang […] möglich ist“*. Wie die Einträge im Einzelnen sortiert sind, ist freigestellt, solange die Informationen so vorliegen, dass sie gezielt gesucht werden können. Dies ist insofern wichtig, als dass Wörterbücher in der Regel nicht von Anfang bis Ende, sondern nur zum Teil gelesen werden (vgl. Schwarze / Wunderlich 1985, S. 369).

Die Informationen sollen für *„einen Bestimmten Benutzer“* zugänglich sein. Somit gibt es verschiedene Wörterbücher für verschiedene Benutzergruppen.

Das Wörterbuch selbst liefert *„bestimmte Informationen“*, diese Informationen können verschiedene Formen annehmen. Es kann zwischen zwei Gruppen von Wörterbüchern unterschieden werden: Solchen, die hauptsächlich Informationen zu den Wörtern an sich geben (Wörterbücher) und solchen, die hauptsächlich Sachinformationen liefern (Lexika). In der Umgangssprache, und auch in der Literatur, werden die Begriffe *Wörterbuch*, *Lexikon* und *Nachschlagewerk* häufig synonym gebraucht und nicht klar voneinander abgegrenzt, auch inhaltlich gibt es Überschneidungen. Lexika liefern auch Informationen zu den behandelten Wörtern und Wörterbücher erklären die behandelten Begriffe (vgl. Schlaefer 2002, S. 77 ; Geeb / Spree, S. 481).

Wörterbücher können weiterhin unterteilt werden in ein- und mehrsprachige Wörterbücher. Die mehrsprachigen lassen sich wiederum unterteilen, in *unidirektionale* Wörterbücher, welche eine Übersetzung von Sprache A nach Sprache B erlauben und *bidirektionale* Wörterbücher, welche auch eine Übersetzung von Sprache B nach Sprache A erlauben (vgl. Atkins 2008, S.39ff). Je nachdem, ob der Benutzer die Sprache spricht, in der er sucht, kann noch zwischen *Hin-* und *Herübersetzungswörterbüchern* unterschieden werden. *Hinübersetzungswörterbücher* sollen helfen ein Wort aus der Muttersprache in eine Fremdsprache zu übersetzen, während *Herübersetzungswörterbücher* aus der Fremd- in die Muttersprache übersetzen sollen (vgl. Schwarze / Wunderlich 1985, S. 377).

Ist im weiteren Verlauf dieser Arbeit von einem Wörterbuch die Rede, so ist damit stets ein (zweisprachiges) (Sprach-)Wörterbuch gemeint, welches Informationen zu Wörtern liefert und als Übersetzungshilfe von einer Sprache in eine andere dienen soll.

1.1.2 **Lexikographische Begriffe**

Wie jede Wissenschaft hat auch die Lexikographie ihre eigenen Begriffe und Fachausdrücke. Einige dieser Begriffe sind unverzichtbar um die Arbeit an *inslex* zu erläutern und es soll genauer betrachtet werden, was diese bedeuten.

Lexem: Ein Lexem, auch *lexikalische Einheit* genannt, bezeichnet eine Einheit des Wortschatzes. Ein Lexem ist häufig mit einem Wort identisch. (vgl. Schwarze / Wunderlich 1985, S. 9) Lexeme können auch aus mehreren Wörtern bestehen, wenn die Bedeutung der Wörter in ihrer Gesamtheit von

der Bedeutung der einzelnen Wörter abweicht (Idiomatisierung). So hat der Ausdruck *„das Herz in der Hose haben“* weder etwas mit dem „Herz“ als Organ, noch mit der „Hose“ als Kleidungsstück zu tun, sondern mit „sich fürchten“ (vgl. Schwarze /Wunderlich 1985, S.9 ; Schlaefer 2002, S.33ff).

Sem, Semem: Ein Sem, oder Bedeutungsmerkmal, beschreibt einzelne Teilbedeutungen zu einem Begriff, etwa *„zum sitzen“* und *„mit Rückenlehne“* für einen Sessel (vgl. Schwarze 1985, S.75), oder *„zylindrisch“* und *„Maschinenteil“* für einen Kolben (vgl. Schlaefer 2002, S.21). Die Seme formen gemeinsam ein Semem zu einem Lexem. Ein Lexem kann dabei aus mehreren Sememen bestehen, für *Kolben* gibt es z. B. die Teilbedeutungen Maschinenteil, Glasgefäß oder Nase (vgl. Schlaefer 2002, S. 20).
Die Unterteilung eines Lexems in mehrere Sememe (auch Bedeutungen oder Lesarten genannt) ist insbesondere bei Übersetzungen, und damit in zweisprachigen Wörterbüchern, wichtig, da unterschiedliche Sememe eines Lexems unterschiedlich übersetzt werden können.

Polysem/Homonym: Mehrdeutige Begriffe werden als Polysem oder Homonym bezeichnet. Beispiele hierfür sind Bank (Sitzgelegenheit und Geldhaus, Homonym) oder Pferd (Tier und Schachfigur, Polysem). Der Unterschied besteht darin, dass es sich bei einem Polysem um ein Lexem mit mehreren Bedeutungen handelt, bei einem Homonym hingegen um mehrere Lexeme, die gleich geschrieben werden. Ob es sich bei einem mehrdeutigen Begriff um ein Polysem oder Homonym handelt hängt unter anderem von der Etymologie (Herkunft) des Wortes ab und ist, nicht immer klar zu trennen. In Wörterbüchern bekommen Homonyme üblicherweise jeweils einen eigenen Artikel wohingegen Polyseme in einem Artikel zusammenstehen (vgl. Schwarze / Wunderlich 1985, S. 53, Schlaefer 2002, S.19).

Lemma: In einem Wörterbuch wird ein Lexem „durch ein Stichwort oder Lemma (griech. *Thema*, *Überschrift*) repräsentiert. Das Lemma bildet mit den zugeordneten Informationen einen Artikel als die „kleinste selbständige Informationseinheit des Wörterbuchs.“ (Schlaefer 2002, S.76) Das Lemma stellt also das Stichwort dar, unter dem der Benutzer im Wörterbuch nachschlägt.
Um an die gewünschte Information zu gelangen, muss es für den Nutzer erkennbar sein unter welchem Lemma diese eingeordnet sein könnte. Da dies bei veränderlichen Wortformen zu Problemen führen könnte werden die

Lemmata zu Nomen üblicherweise als Nominativ Singular, zu Verben als Infinitiv und zu Adjektiven in der unflektierten Form angesetzt (vgl. Bergenholtz / Tarp 1995, S. 15). Aus mehreren Wörtern bestehende Lemmata werden unter dem ersten oder wichtigsten Wort angesetzt (vgl. Schlaefer 2002, S.88).
Welche Lexeme als Lemmata gewählt werden und wie verschiedenen Varianteneines Lemmata umgegangen wird ist ein eigener Forschungsgegenstand (vgl. Schlaefer 2002, S.88f). Für *inslex* ist der Vorgang der Lemmatisierung bereits abgeschlossen und soll nicht Thema dieser Arbeit sein.

1.1.3 **Allgemeine Wörterbuchstruktur**

So unterschiedlich Wörterbücher auch sein mögen, folgen sie doch alle einer gewissen Struktur. Diese Struktur lässt sich Nutzen um Wörterbücher näher zu beschreiben.

Ein Wörterbuch gliedert sich in zwei große Teile, einen *Artikelteil*, worin die eigentlichen Informationen stehen und sie sogenannten *komplementären Wörterbuchteile*, worin Informationen über das Wörterbuch zu finden sind, wie etwa das Vorwort, Informationen zur Lemmaauswahl oder ein Abkürzungsverzeichnis (vgl. Schlaefer 2002, S.83). Der Artikelteil gliedert sich wiederum in eine *Makrostruktur*, welche die Reihenfolge der Artikel beschreibt und eine *Mikrostruktur*, welche die Gestaltung der Informationen innerhalb der Artikel beschreibt (vgl. Carstensen et al. 2004, S.436).

Die komplementären Wörterbuchteile sind in der Regel klar vom Artikelteil abgegrenzt, z. B. als Vorwort und Anhang. Die Anordnung von Artikelteil und komplementären Teilen wird durch die *Rahmenstruktur* festgelegt (vgl. Bergenholtz / Tarp 1995, S.15, 211ff). Desweitern lässt sich die Struktur eines Wörterbuches über die äußere und innere *Zugriffsstruktur* beschreiben. Die äußere Zugriffsstruktur soll den Benutzer zum richtigen Artikel im richtigen Wörterbuch führen (z. B. über eine entsprechende Einbandgestaltung), die innere Zugriffsstruktur soll es dem Benutzer erleichtern die gewünschte Information innerhalb des Artikels zu finden (z. B. über unterschiedliche typografische Gestaltung der verschiedenen Artikelteile) (vgl. Bergenholtz / Tarp 1995, S. 15f, 219ff). Die Art und Weise wie auf weitere Informationen (inner- und außerhalb) des Lexikons Verwiesen wird nennt sich *Verweisstruktur* (vgl. Bergenholtz / Tarp 1995, S. 15, 215ff). Für die Gestaltung der

inslex Datenbank sind besonders die Makro- und Mikrostruktur von Belang und sollen deshalb näher betrachtet werden.

1.1.3.1 **Makrostruktur**

Der zentrale Punkt der Makrostruktur ist die Reihenfolge, in welcher die Einträge sortiert sind. Sie bestimmt an welcher Stelle ein Lemma im Artikelteil steht (vgl. Bergenholtz / Nielsen / Tarp 2009, S.317). Da die Informationen in einem Wörterbuch gezielt gefunden werden sollen, ist es sehr wichtig die Artikel so anzuordnen, dass der Nutzer aus dem Lemma selbst auf seine Position im Wörterbuch schließen kann. Für die Ordnung der Artikel existieren zwei grundlegende Prinzipien: Nach dem Alphabet oder thematisch.

Die meisten Wörterbücher sind alphabetisch sortiert (vgl. Bergenholtz / Nielsen / Tarp, S.330 ; Schwarze / Wunderlich 1985, S. 369). Das zugrunde liegende Prinzip besteht darin, dass die Buchstaben eines Alphabets in einer festen Reihenfolge vorliegen. Ist diese Reihenfolge dem Benutzer bekannt, kann er aus der Schreibweise des Lemmas auf die Position des Artikels schließen (vgl. Bergenholtz / Tarp 1995, S. 190). Wie die einzelnen Artikel genau angeordnet werden, variiert dennoch von Wörterbuch zu Wörterbuch, auch wenn dasselbe Alphabet verwendet wird. So gibt es verschiedene Ansätze wie mit Lemmata mit Leerzeichen oder Bindestrichen oder den deutschen Umlauten verfahren wird (vgl. Bergenholtz / Tarp 1995, S.191ff).

Für die alphabetische Sortierung gibt es zwei verschiedene Ansätze. Bei der glattalphabetischen Sortierung werden die Lemmata streng nach ihrer Schreibweise angeordnet. Um Platz zu sparen und Zusammenhänge zwischen Lemmata aufzuzeigen werden die Lemmata teilweise in einer nestalphabetischen Struktur aufgelistet. Hierbei bleibt die eigentliche alphabetische Sortierung erhalten, folgt einem Lemma ein etymologisch zusammenhängender Wortausschnitt, so wird dieses Lemma als Leitlemma angesehen und ihm die restlichen Sublemmata, häufig eingerückt, angehängt. (vgl. Schlaefer 2002, S.89f ; Bergenholtz / Tarp 1995, S. 192ff).
Die nestalphabetische Sortierung ordnet alle Lemmata eines Wortstammes einem Hauptlemma unter. Sie stehen alle in einem *Nest*, unabhängig von der Schreibweise der einzelnen Lemmata. Um den gewünschten Artikel zu finden, muss unter dem Eingangslemma gesucht werden, so ist beispielsweise der Artikel zu *„Schwarzerle“* in einer nestalphabetischen Sortierung unter

dem Lemma *„Erle“* zu finden (vgl. Schlaefer 2002, S.89f ; Bergenholtz / Tarp 1995, S. 194).

Die thematische Sortierung ordnet setzt die einzelnen Lemmata in eine semantische Beziehung zueinander, etwa indem mehrere Lemmata einem gemeinsamen Oberbegriff zugeordnet werden, die Ordnung basiert hierbei auf den Bedeutungen der Lemmata, unabhängig von der Schreibweise (vgl. Bergenholtz / Tarp 1995, S. 195). Diese Struktur ist mit einem Baum zu vergleichen, der sich immer weiter verzweigt, von allgemeinen Oberbegriffen hin zu immer präziseren Unterbegriffen (vgl. Carstensen et al. 2004, S. 435). Innerhalb der Baumstruktur sind Lemmata einer Ebene in der Regel nach dem Alphabet geordnet, so dass meistens keine reine systematische Ordnung, sondern eine Kombination aus systematischer und alphabetischer Ordnung vorliegt.

Die Systematische Sortierung wird hauptsächlich in Bereichen angewendet, in denen eine Organisation nach Themengebieten bereits im Thema selbst zugrunde liegt, z. B. in der Medizin oder den Naturwissenschaften, wo sich ein Begriff recht eindeutig einem Gebiet zuordnen lässt (vgl. Bergenholz / Tarp 1995, S. 199). In Wörterbüchern, die sich speziell mit kulturabhängigen Themen wie Politik oder Wirtschaft beschäftigen, sind die Trennungen zwischen den Bereichen häufig unscharf und thematische Sortierungen werden nur selten angewendet (vgl. Geeb / Spree 2004, S. 478).

Der Vorteil der systematischen Sortierung liegt hauptsächlich darin, dass die Lemmata im jeweiligen Kontext einsortiert werden. Dies ermöglicht es dem Nutzer einen Überblick, über das jeweilige Themengebiet zu bekommen. Sowohl inhaltlich über das Thema, als auch was die Abdeckung des Themas durch das Wörterbuch betrifft. Insbesondere bei Übersetzungen kann es hilfreich sein, das thematische Umfeld eines Lemmas mit einzubeziehen, sei es weil so gleich mehrere Übersetzungsfragen auf einmal gelöst werden können, sei es weil bestimmte Übersetzungsfragen einen gewissen Überblick über den Themenkomplex verlangen. Außerdem ist es möglich einen Wörterbucheintrag zu finden, ohne die genaue Schreibweise des gesuchten Begriffes zu kennen. (vgl. Bergenholtz / Tarp 1995, S. 199 ; Geeb / Spree 2004, S. 478).

Ein Nachteil einer solchen Makrostruktur liegt darin, dass der Benutzer soweit mit dem Thema vertraut sein muss, dass er in der Lage ist das Lemma

an der richtigen Stelle zu suchen. Aus diesem Grund verfügen thematisch sortierte Wörterbücher in der Regel über einen alphabetischen Index, der das Auffinden der Lemmata erleichtern soll (vgl. Bergenholtz / Tarp 1995, S. 198f). Desweiteren gibt es, anders als beim Alphabet, keine natürliche Ordnung. Die Systematik, wonach die Lemmata einander Zugeordnet werden, kann frei bestimmt werden. Derselbe Begriff kann in verschiedenen Wörterbüchern unterschiedlich eingeordnet werden, je nachdem nach welchen Gesichtspunkten die systematische Ordnung erstellt wurde (vgl. Schlaefer 2002, S.91f).

Dieses Problem stellt sich nicht bei einer Sortierung nach dem Alphabet. Ist das System nachdem eventuelle Sonderfälle angesetzt werden dem Nutzer bekannt, kann er einfach innerhalb des Wörterbuches navigieren.

1.1.3.2 **Mikrostruktur**

Die Mikrostruktur beschreibt den Aufbau der Artikel. Sie bestimmt welche Informationen enthalten, wie diese Angeordnet sind und wie dies dem Nutzer vermittelt werden soll. Ein Artikel besteht in der Regel aus zwei Teilen, einem Teil, der das Lemma an sich beschreibt (z. B. Informationen zur Silbentrennung) und einem Informationsteil, der die Informationen zum Lemma enthält (z. B. Erklärung und Übersetzung). Die Abgrenzung dazwischen erfolgt durch gestalterische Elemente, etwa indem das Lemma fett gedruckt wird (vgl. Schlaefer 2002, S. 85f).

Die Mikrostruktur eines Artikels lässt sich als eine Sammlung von Datenfeldern betrachten, wobei jedes Feld eine ganz bestimmte Rolle erfüllt. Es gibt beispielsweise Felder, die das Lemma definieren, die Aussprache angeben, unregelmäßige Formen erläutern, oder die Entsprechung des Lemmas in einer anderen Sprache angeben, wobei nicht jeder Feldtyp in jedem Artikel vorkommen muss (vgl. Bergenholtz / Tarp 1995, S.200 ; Schlaefer 2002, S.87). Eine umfangreiche Aufstellung von möglichen Feldtypen findet sich bei Bergenholtz / Nielsen / Tarp 2009, S. 333ff und Atkins 2008, S.203ff. Die einzelnen Felder werden durch typographische Elemente definiert, etwa durch Klammern oder kursiven Text. Zusätzlich spiegelt die Position des Feldes innerhalb des Artikels seine Funktion wider. Dies ermöglicht es dem Nutzer aus dem generellem Artikelaufbau und der typographischen Gestaltung auf die Funktion des jeweiligen Teils des Artikels zu schließen (vgl. Schlaefer 2002, S.87f). Die Artikel sind möglichst so gestaltet, dass es dem

Nutzer einfach gemacht wird sich auch in längeren Abschnitten zurechtzufinden und gezielt Informationen zu finden, ohne den kompletten Artikel lesen zu müssen (vgl. Atkins 2009, S. 203). Verschiedene Wörterbücher verfolgen verschiedene Ansätze bei der Mikrostruktur, innerhalb eines Wörterbuches folgt sie dem gleichen Prinzip.

1.1.4 **Onlinewörterbücher**

Bei einem Onlinewörterbuch handelt es sich um ein elektronisches Wörterbuch, welches über das Internet zugänglich ist. Die bereits angesprochenen Aspekte treffen auch auf Onlinewörterbücher zu. Da es sich bei *inslex* um ein Onlinewörterbuch handelt, soll auf einige der besonderen Eigenschaften dieser Wörterbuchform eingegangen werden.

Die Rahmenstruktur eines Onlinewörterbuches wird durch die umgebende Webseite gebildet. Hierüber ist der Zugriff auf das Wörterbuch möglich, auch die komplementären Wörterbuchteile sind auf der gleichen Seite zu finden. Die angebotenen Möglichkeiten einen Artikel im Wörterbuch zu suchen stellen die Makrostruktur dar, etwa über eine Suchmaske, oder eine der Printversion nachempfundenen Auflistung der Lemmata. Die Mikrostruktur eines Onlinewörterbuches spiegelt sich in der konkreten Anzeige der Artikel auf dem Bildschirm wider.

Ein Wörterbuch online zu publizieren eröffnet einige neue Möglichkeiten, darunter fallen (vgl. Lemberg 2001, S. 73ff):

- Der Umfang des Wörterbuches ist nicht mehr durch die physikalische Form beschränkt, es können nahezu beliebig viele und lange Artikel online gestellt werden. Dies hat auch zu Folge, dass viele lexikographische Verdichtungsformen (wie Abkürzungen und Auslassungen) nicht mehr notwendig sind. Die Mikrostruktur lässt sich so häufig ansprechender präsentieren, als dies in gedruckter Form auf engem Raum möglich ist.
- Das Wörterbuch kann auf vielfältige Arten durchsucht werden. Die Artikel können je nach Bedarf neu angeordnet werden. Es ist nicht mehr notwendig sich bei der Erstellung des Wörterbuches auf eine Sortierung festzulegen. Artikel können nicht nur über die Lemmata, sonder auch über eine Volltextsuche gefunden werden. Die Makrostruktur ist somit nicht festgefügt, sondern dynamisch.

- Verweise zwischen Artikeln und zu externen Quellen können als Hyperlink realisiert werden. Dies vereinfacht es, diesen Verweisen zu folgen.
- Es ist möglich das Wörterbuch um andere Medien (wie Video oder Audio) zu ergänzen Auch Bilder oder Diagramme lassen sich einfügen, ohne auf den verfügbaren Druckumfang Rücksicht nehmen zu müssen.
- Das Wörterbuch kann jederzeit und ohne viel Aufwand ergänzt und aktualisiert werden. Es ist nicht notwendig Korrekturen und Ergänzungen zu sammeln und sie als neue Auflage herauszubringen, sondern sie können direkt und mit geringer Verzögerung in das Wörterbuch eingefügt werden.

1.2 Niederdeutsche Sprache

Beim Niederdeutschen handelt es sich um eine in Norddeutschland gesprochene Sprachvariante des Hochdeutschen. Ob es sich um eine eigene Sprache (vgl. Bargstedt 2009, S. 47) handelt, oder ob das Niederdeutsche ein Dialekt des Hochdeutschen ist (vgl. Knoop 1997, S. 36f), ist umstritten. Da es sich wiederum in einzelne, sich z.T. stark voneinander unterscheidende, Dialektareale gliedert (vgl. Lindow et al. 1998, S. 19), wird ihm zumindest eine Sonderstellung im deutschen Sprachraum zugestanden (vgl. Knoop 1997, S. 36).

Abbildung 1: Dialektgebiete der Bundesrepublik Deutschland (Knoop 1997, S18 ; vgl. auch Bargstedt 2009, S. 48f)

„Platt“ im Sinne von *Plattdeutsch* bedeutet soviel wie „einfach, deutlich, klar“ (vgl. Bargstedt 2009, S.47). Die Begriffe *Niederdeutsch* und *Plattdeutsch* werden häufig synonym gebraucht. Beispielsweise führt das Institut für *niederdeutsche* Sprache Veranstaltungen zum Thema „*Plattdeutsch* gestern und morgen“ durch (vgl. Zurawski 2007, Titel), ein *Plattdeutsches* Wörterbuch betitelt den Artikelteil „*Niederdeutsch* – Hochdeutsch“ (vgl. Fehrs-Gilde 2004, S.21) und ein Werk mit dem Titel „*Platt!*“ beschreibt „die Sprache *Niederdeutsch*“ (vgl. Bargstedt 2009, S. 47).
Da die bisherige Version von *inslex* den Begriff *„Plattdeutsch“* für die nicht-hochdeutschen Begriffe benutzt, wird dieser Ausdruck im Weiteren verwendet.

Das älteste Plattdeutsch stammt aus dem 9. Jahrhundert, seinen höchsten Verbreitungsgrad erreichte es, mit der Hanse, als Mittelniederdeutsch im 14. bis 16. Jahrhundert. Zu diesem Zeitpunkt wurde es im norddeutschen Raum sowohl gesprochen, als auch geschrieben und war international verbreitet.

Im 16. Jahrhundert verlor das Plattdeutsche seinen Status als Schriftsprache, durch den Niedergang der Hanse und das zunehmende Übergewicht des Hochdeutschen. Platt wurde seitdem fast nicht mehr geschrieben, sondern nur noch gesprochen, meistens auf dem Land und von unteren sozialen Schichten (vgl. Bargstedt 2009, S. 125ff).

Eine Rückbesinnung auf das Plattdeutsche auch in der Literatur setzte ab dem 19. Jahrhundert ein (vgl. Lindow et al. 1998, S.17). Heute ist das Plattdeutsche in Zahlreichen Lebens- und Kulturbereichen vertreten. Es gibt Plattdeutsches Radio, Fernsehen, Theater und Musik, wird allerdings häufiger gesprochen als geschrieben (vgl. Lindow et al. 1998, S. 19 ; Bargstedt 2009, S.143, 187ff). Im Internet gibt es zahlreiche Plattdeutsche oder zweisprachige Seiten, die sich sowohl mit der Sprache an sich, als auch mit anderen Themen auf Plattdeutsch beschäftigen (vgl. Zurawski 2007, S. 154ff). Plattdeutsch ist in Schleswig-Holstein als Amtssprache anerkannt und steht in der „EU-Charta für Regional- und Minderheitensprachen“ (vgl. Bargstedt 2009, S. 143).

Ein Anspruch einer solchen lebendigen Sprache ist es möglichst alles darin ausdrücken zu können und sie in allen Lebensbereichen anwenden zu können. Auch wenn das Plattdeutsche häufig in der Familie und im informellen Kontext verwendet wird, soll es möglich sein sich darin über sämtliche Themenbereiche zu unterhalten. Ein Aspekt hiervon ist es, dass sich Sprache permanent verändert und neue Ausdrücke generiert werden (vgl. Bargstedt 2009, S. 187ff). Bargstedt (2009) spricht in diesem Zusammenhang von einer „sprachlichen Bedarfsdeckung“ (S. 191). Die Sammlung von plattdeutschen Ausdrücken in *inslex* dient dieser Bedarfsdeckung und soll helfen das Plattdeutsche lebendig zu halten.

Auf eine linguistische Definition des Plattdeutschen sei hier aus Platzgründen verzichtet. Für die Entwicklung der *inslex* Datenbank spielen die genauen grammatikalischen Unterschiede zwischen Hoch- und Plattdeutsch keine Rolle. Weitere Informationen zur Entwicklung und Abgrenzung des Plattdeutschen, wie die *Nichtteilhabe an der hochdeutschen Lautverschiebung*, oder die *Erhaltung der germanischen Verschlusslaute*, finden sich bei Lindow et al. (1998, S. 17ff) und Knoop (1997, S. 25ff).

1.3 Das Institut für niederdeutsche Sprache

Beim Institut für niederdeutsche Sprache (INS) handelt es sich um einen, 1973 gegründeten, gemeinnützigen Verein mit Sitz in Bremen. Die Ziele des Vereins sind laut Satzung:

> *a. Sammlung, Ordnung und wissenschaftliche Analyse von niederdeutschen Spracherzeugnissen mit besonderer Berücksichtigung der Gegenwart,*
>
> *b. Aufbereitung der Arbeitsergebnisse für die Öffentlichkeit,*
>
> *c. Koordination und Unterstützung aller Bemühungen um die niederdeutsche Sprache einschließlich des Spracherwerbs*
>
> *d. Kontaktpflege mit ähnlichen Institutionen, auch außerhalb der Staatsgrenzen.*
> (INS Satzung 2005)

Das INS arbeitet auf wissenschaftlicher Basis daran die Belange des Niederdeutschen zu dokumentieren und zu analysieren. Es wird staatlich gefördert und arbeitet überregional an der „Pflege und Förderung der niederdeutschen Sprache, Wissenschaft und Kultur“ (INS Homepage 2011).

Um dies zu erreichen unterhält das INS unter anderem eine eigene Bibliothek, in der neben Primär- und Sekundärliteratur auch Bühnenmanuskripte, Kirchen- und Schulliteratur, sowie regionale Zeitschriften gesammelt werden. Das INS unterstützt zahlreiche Initiativen, um das Plattdeutsche in die Öffentlichkeit zu bringen, wie etwa eine Suche nach dem plattdeutschen „Wort des Jahres“. Auf der Webseite des INS werden Informationen zur plattdeutschen Sprache, Geschichte und Kultur bereitgestellt. Zusätzlich gibt es Übersichten über plattdeutsche Rundfunksendungen, Theaterstücke, Zeitschriften und Veranstaltungen (vgl. INS Homepage 2011).

1.4 *Inslex* – die bisherige Version

Die bisherige Version von *inslex* ist über die Webseite des INS (INS Inslex 2011) zugänglich. Die Suchmöglichkeiten (die Makrostruktur) des Wörterbuches bestehen aus einem Suchfeld und einer Liste der Lemmata. Die Liste ist nach den hochdeutschen Lemmata alphabetisch sortiert, wobei alle groß geschriebenen Lemmata am Anfang der Liste stehen, ist der Buchstabe *„Z“* erreicht wird die Liste mit *„a“* fortgesetzt. Zu jedem Eintrag gibt es eine Schaltfläche *„Details“*, diese zeigt den gesamten Artikel an. Das Blättern in der Liste ist durch Links am unteren Ende möglich. Hierüber kann eine oder fünf Seiten vor/zurück gegangen werden. Größere Sprünge, oder das direkte

Anwählen eines Buchstabens ist nicht möglich.
Über die Sucheingabe lässt sich die Liste filtern, in den Standardeinstellungen wird sowohl der hoch- als auch der plattdeutsche Teil links- und rechtstrunkiert durchsucht. Über die erweiterten Einstellungen, lässt sich die Suche auf eine Sprache einschränken. Nach einer Suche wird die Liste auf die Einträge beschränkt, die den Suchkriterien entsprechen. Da in dieser Ansicht nur der Artikelanfang sichtbar ist, ist der Zusammenhang zwischen Suchwort und Treffer nicht immer sofort ersichtlich. Um zur Sucheingabe

Abbildung 2: Inslex, bisherige Version, Startansicht (INS inslex 2011)

zurückzukehren, kann auf den Button *„Alle Zeigen“* geklickt werden, dies macht die Filterung der Liste rückgängig und blendet das Suchfeld wieder ein.

Die Artikelansicht (Mikrostruktur) besteht aus zwei Feldern, *Hochdeutsch* und *Plattdeutsch*. Im Feld *Hochdeutsch* ist das jeweilige hochdeutsche Lemma eingetragen. Darunter befindet sich das Feld *Plattdeutsch*, hierin stehen die Informationen zum Lemma. Diese bestehen aus den Übersetzungen des Lemmas. Anmerkungen zu den Übersetzungen stehen in Klammern dahinter, gibt es mehrere Übersetzungen sind sie üblicherweise durch Schrägstriche voneinander getrennt. Die Artikelansicht, kann über den Link *„zurück“* verlassen werden. Dieser führt zurück zur Liste, bzw. zu den gefilterten Listeneinträgen.

Abbildung 3: Inslex, bisherige Version, nach „aufnehmen“ gefilterte Liste (INS inslex 2011)

Abbildung 4: Inslex, bisherige Version, Detailansicht zu „aufnehmen“ (INS inslex 2011)

1.4.1 **Datenherkunft**

Der Norddeutsche Rundfunk (NDR) sendet regelmäßig Nachrichten auf Plattdeutsch. Zu diesem Zweck werden Nachrichtenmeldungen des NDR vom Hochdeutschen ins Plattdeutsche übersetzt. Die hierbei anfallenden Übersetzungen bilden die Grundlage für die in *inslex* gespeicherten Daten. Da die Einträge aus aktuellen Nachrichtenmeldungen stammen finden sich darunter viele Wörter, die nicht in andern plattdeutschen Wörterbüchern zu finden sind. Hierunter fallen insbesondere zeitgenössische Begriffe und spezielle Wortzusammensetzungen. Wird im Zuge der allgemeinen Sprachentwicklung ein neuer hochdeutscher Begriff geprägt, so muss eine plattdeutsche Entsprechung gefunden werden. *Inslex* erhebt keinen Anspruch auf Vollständigkeit. Es soll nicht die gesamte plattdeutsche Sprache abgebildet werden, sondern nur der Teil der NDR-Meldungen. Da der Fokus auf der Übersetzung eines bestimmtem Ausdrucks im Kontext der jeweiligen Meldung liegt, wird auf weiterführende Informationen verzichtet, wie etwa auf Angaben zur Grammatik oder zur Herkunft des Begriffes. Auch werden nicht alle Übersetzungsmöglichkeiten aufgeführt, sondern nur die in der jeweiligen Situation verwendeten. *Inslex* soll die praktische Arbeit unterstützen und weniger ein Forschungsprojekt sein.

Wird eine neue Meldung übersetzt, werden interessante Ausdrücke von den jeweiligen Redakteuren auf ihr Vorhandensein in *inslex* überprüft und gegebenenfalls neu hinzugefügt. Die Beiträge stammen größtenteils aus dem Hamburger und Bremer Sprachraum und werden gemäß den Rechtschreibregeln von Johannes Saß eingegeben.

1.4.2 **Zielgruppe**

Die Zielgruppe von *inslex* stellen zum einen die an der Übersetzung und Verwendung der Nachrichtenmeldungen beteiligten Personen dar. Hier dient *inslex* als Werkzeug, um die Übersetzung zu unterstützen, indem ge-

prüft werden kann wie ein Begriff in der Vergangenheit übersetzt wurde. Zum anderen zielt *inslex* auf interessierte Laien, aus diesem Grund ist die Datenbank auch öffentlich über die Webseite des INS zugänglich.

Beiden Gruppen ist gemein, dass Grundkenntnisse im Plattdeutschen vorhanden sind. Diese Grundkenntnisse sind auch erforderlich um die Angaben in *inslex* richtig zu interpretieren, etwa welche Übersetzung in welchem Fall in Frage kommt.

1.4.3 **Datenbestand**

Der Datenbestand für die *inslex* Datenbank umfasst momentan 5.000 Einträge (hochdeutsche Lemmata). Es ist mir einem Zuwachs von ca. 1.000 Einträgen pro Jahr zu rechnen, so dass die Datenbank mittelfristig 10.000 Einträge umfassen wird. Der Datenbestand ist im Laufe der vergangenen Jahre immer weiter gewachsen, eine geplante Lemmaauswahl im Sinne herkömmlicher Wörterbücher hat nicht stattgefunden. Die Einträge sind dann hinzugefügt worden, wenn sie in einer der NDR Meldungen verwendet wurden, auch erfolgte das Anlegen neuer Begriffe durch wechselnde Personen.

Dies hat zur Folge, dass die Einträge in *inslex* nicht alle demselben Schema folgen. Bei der Durchsicht der vorhandenen Einträge konnten die folgenden Punkte ermittelt werden:

- Die hochdeutschen Begriffe bestehen fast immer aus einem Wort.
- Bei den plattdeutschen Begriffen sind Zusammensetzungen aus mehreren Wörtern nicht selten
- Zu einem hochdeutschen Begriff gibt es teilweise mehrere plattdeutsche Entsprechungen.
- Zum Teil werden die plattdeutschen Begriffe näher erklärt, bzw. auf ein bestimmtest Gebiet beschränkt, indem der jeweilige Kontext in Klammern dahintersteht.
- Bei den plattdeutschen Begriffen finden sich auch ganze Sätze, oder Satzteile, die jeweils von einer hochdeutschen Übersetzung gefolgt werden.

Ausgehend von diesen Punkten können allgemeine Aussagen zum Datenbestand getroffen werden:

- Es gibt mehrere Möglichkeiten einen hochdeutschen Begriff ins Plattdeutsche zu übersetzen
- Gleich lautende hochdeutsche Begriffe mit unterschiedlichen Bedeutungen (Homonyme) haben teils unterschiedliche plattdeutsche Übersetzungen.
- Wird ein hochdeutsches Wort in einem bestimmten Kontext verwendet kann die plattdeutsche Übersetzung stark von der regulären Übersetzung abweichen. Die wörtliche Übersetzung des Begriffes kommt nicht zwangsläufig in der Übersetzung vor. So wird *ausdehnen* mit *„utwieden“* übersetzt. Wird *ausdehnen* jedoch im Kontext von *„einen Streik ausdehnen“* benutzt, so lautet die Übersetzung „*in anner Flachen rindrägen“*.

Für die Organisation der Datenbank hat dies zur Folge, dass erstens zwischen hoch- und plattdeutschen Begriffen unterschieden werden muss. Zweitens müssen Homonyme und Polyseme je nach Bedeutung unterschieden werden. Drittens müssen abweichende Übersetzungen eines Wortes angegeben werden, wenn dieses Wort in einem Bestimmten Kontext verwendet wird und Beispielanwendungen gegeben werden.

1.5 Konzept für die neue Version

Der nächste Schritt bei der Erstellung der Neuauflage von *inslex* bestand darin, den vorhandenen Datenbestand in eine lexikographische Form zu bringen. Hier galt es vor allem eine geeignete Rahmen-, Makro- und Mikrostruktur zu entwerfen. Auf Basis dieser Strukturen soll die eigentliche Datenbank erstellt werden.

1.5.1 Funktionsumfang

Bevor mit der eigentlichen Entwicklung der Datenbank begonnen wurde, wurden die Anforderungen und Wünsche des INS ermittelt. Hierbei wurden die folgenden Punkte erarbeitet:

- Die Daten sollen über ein Eingabefeld durchsucht werden können. Hierbei ist es wünschenswert nur einen Teil des Wortes eingeben zu müssen, um das gesuchte Wort zu finden (trunkierte Suche).
- Zusätzlich zum Suchfeld soll dem Benutzer eine alphabetisch sortierte Liste der hochdeutschen Lemmata zur Verfügung stehen.

- Es wird erwartet, dass die Anfragen zu ca. 90% eine plattdeutsche Übersetzung für einen hochdeutschen Begriff suchen. Die Suche nach Übersetzungen soll dennoch auch hochdeutsche Übersetzungen zu plattdeutschen Suchbegriffen finden.
- Die Reihenfolge der Anzeige der plattdeutschen Übersetzungen soll bei der Dateneingabe frei festlegbar sein.
- Einmal angelegte Einträge sollen veränder- und erweiterbar sein.
- Es ist nur mit wenigen Benutzern gleichzeitig zu rechnen. Auf das Back-End des Systems sollen insgesamt ca. zehn Personen Zugriff haben. Die Nutzerzahlen für das Front-End sind nur schwer einzuschätzen, es ist hier allerdings nicht mit großen Mengen zu rechnen.
- Die Datenbank soll online zugänglich sein.
- Die Datenbank soll in die bestehende Webseite des INS eingebunden werden. Die Seite wird mit dem Content-Management-System Typo3 betrieben. Der Webserver unterstützt PHP und MySQL.

1.5.2 **Rahmenstruktur**

Der Zugriff auf das Wörterbuch erfolgt über die Webseite des INS (http://www.ins-bremen.de). Es gliedert sich in zwei Unterseiten, eine enthält die Suchmöglichkeit und die Anzeige der Ergebnisse, eine die komplementären Wörterbuchteile. Zusätzlich gibt es einen nicht öffentlichen Bereich (Back-End) über den die Verwaltung des Wörterbuches erfolgt.

1.5.3 **Makrostruktur**

Bei der Makrostruktur müssen zwei Ebenen unterschieden werden. Eine Datenbankebene, welche bestimmt wie die Einträge innerhalb der Datenbank organisiert werden und eine Benutzerebene, auf der die Einträge für den Benutzer angezeigt werden. Die Organisation der Datenbank wird in Abschnitt 3.1 beschrieben, sie besteht aus mehreren Tabellen, die miteinander verknüpft sind. Die Reihenfolge der Einträge in der Datenbank hat keinen Einfluss auf die Sortierung der Ausgabe, das Erzeugen der Makrostruktur auf der Benutzerebne erfolgt bei der Ausgabe des Datenbankinhaltes. Zusätzlich zur Suchmöglichkeit ist hier eine alphabetisch sortierte Liste der hochdeutschen Lemmata zu nennen. Eine genaue Beschreibung der einzelnen Elemente findet sich in Abschnitt 2.1.

Der Fokus des Wörterbuches liegt auf der Übersetzungsrichtung Hochdeutsch – Plattdeutsch. Aus diesem Grund ist die Makro- und Mikrostruktur vom Hochdeutschen ausgehend gestaltet, ähnlich wie unidirektionales Wörterbuch (da die Erläuterungen ausschließlich in Hochdeutsch gegeben werden, könnte man soweit gehen *inslex* als ein rein unidirektionales Wörterbuch zu bezeichnen). Eine Suche nach plattdeutschen Begriffen ist technisch ohne weiteres möglich, die Artikel behalten ihre Struktur jedoch unabhängig von der Übersetzungsrichtung bei. Sofern nicht anders angegeben bezeichnet *Lemma* aus diesem Grund das hochdeutsche Artikelstichwort und *Übersetzung* die plattdeutsche Entsprechung. Wird nach einem plattdeutschen Begriff gesucht, so werden die gefundenen Artikel so ausgegeben, als wäre direkt eine Suche nach den hochdeutschen Begriffen erfolgt. Hat ein plattdeutscher Begriff mehrere Übersetzungen werden demnach mehrere Artikel angezeigt.

1.5.4 **Mikrostruktur**

Wie in Abschnitt 1.1.3.2 beschrieben besteht die Mikrostruktur eines Wörterbuchartikels aus einzelnen Feldern, die jeweils mit bestimmten Inhalten gefüllt werden. Ausgehend vom Datenbestand konnten die folgenden Felder ermittelt werden:

Lemma: Hierin steht das hochdeutsche Lemma. Üblicherweise ist dies der Begriff nachdem gesucht wurde. Dieses Feld ist in jedem Artikel genau ein Mal vorhanden.

Übersetzung: Hierin stehen die plattdeutschen Lemmata. (Die Artikel werden immer vom Hochdeutschen ausgehend ausgegeben, ob es sich bei den plattdeutschen Begriffen deshalb um Lemmata im streng lexikographischen Sinn handelt könnte angezweifelt werden. Da die plattdeutschen Begriffe bei einer Suche jedoch die Funktion eines Lemmas erfüllen, soll dieser Begriff verwendet werden.) Es können mehrere Übersetzungen zu einem Lemma vorhanden sein. Gibt es keine direkte Übersetzung zu einem Lemma, dafür aber Phrasen (s.u.) kann dieses Feld frei bleiben.

Phrase: Dieses Feld beinhaltet feststehende Redewendungen, sowie Anwendungsbeispiel. Zum Teil kommt es vor, dass ein Lemma in einem bestimmten Zusammenhang anders übersetzt wird als die wörtliche Übersetzung. Hat ein Lemma keine wörtliche Übersetzung, sondern nur einen spezifischen Kontext in dem es übersetzt wurde, ist es möglich, dass nur ei-

ne Phrase, und keine Übersetzung, angegeben wird. Ist dieses Feld gefüllt, so muss auch eine Übersetzung dazu angegeben werden.

Übersetzung der Phrase: Hierin steht die Übersetzung der Phrase.

Erläuterungen: *Inslex* beinhaltet primär Übersetzungen zu Begriffen. Zusätzlich enthält es jedoch weitere Informationen zu den Begriffen selbst. Hierzu existiert zu jedem der Felder ein zusätzliches Feld für diese Angaben. Der genaue Inhalt des Erläuterungsfeldes hängt vom zugehörigen Hauptfeld ab und kann von Artikel zu Artikel variieren. Die Erläuterungen zu den Lemmata bestehen hauptsächlich aus der Unterscheidung vom Homonymen. Existieren mehrere gleichgeschriebene Lemmata mit unterschiedlichen Bedeutungen, gibt das Erläuterungsfeld Aufschluss darüber, welche gemeint ist. Die Erläuterungen zu den plattdeutschen Begriffen enthalten häufig Angaben zum Wort selbst, wie etwas unregelmäßige Formen, oder besondere Aussprachen. Auch können hier Angaben gemacht werden, wenn ein Ausdruck nur in einem bestimmten Kontext zu gebrauchen ist. Die Erläuterungen zu den Phrasen ähneln denen der Übersetzungen, enthalten aber auch wörtliche Übersetzungen von Redewendungen. Die Angabe von Erläuterungen muss nicht zu jedem Feld erfolgen, sie ist jedoch nur dann zulässig, wenn das zu erläuternde Feld auch ausgefüllt ist.

In der Erläuterung können sehr unterschiedliche Daten gespeichert werden. Bei einem Wort mag dies eine Angabe zur Herkunft (*„aus der Seemannssprache“*) sein, bei einem Anderen ein Hinweis zur Verwendung (*„meistens im Plural“*). Eine Aufteilung der Erläuterungen in mehrere Felder, etwa für Grammatik, Etymologie oder Bedeutungsunterscheidung war angedacht. Allerdings konnten bei der Durchsicht der Daten kaum Begriffe gefunden werden, zu denen mehrere Erläuterungen angegeben werden. Dies hätte zu einer großen Anzahl an leeren Feldern im Artikel geführt, eine Suche innerhalb der Erläuterungen hätte aber dennoch alle Felder durchsuchen müssen. Zudem hätte dies die Dateneingabe unnötig verkompliziert, da sich der Benutzer mit zahlreichen nicht benötigten Feldern konfrontiert sähe. Da die Artikel unterschiedlich beschaffen sind und nicht bei jedem alle Erläuterungstypen vorkommen, wären Fehleingaben (ein Erläuterungstext wir einem Falschen Feld zugeordnet) wahrscheinlich gewesen. Darüberhinaus stellen Eingaben mit mehreren Angaben in einem Erläuterungsfeld (*„aus der Seemannssprache / meist im Plural“*) weder bei der Dateneingabe, noch

bei der Suche ein Problem dar (die Erläuterungen werden im Volltext durchsucht).

Die optische Gestaltung der Mikrostruktur basiert auf zwei Spalten, jeweils für die hoch- und plattdeutschen Begriffe. Die hochdeutschen Begriffe stehen dabei auf der linken Seite. Ein Artikel besteht aus zwei horizontalen Abschnitten, im oberen finden sich die Lemmata, im unteren die Phrasen. Typographisch werden die Phrasen in einer kleineren Schriftart als die Lemmata dargestellt. Die Erläuterungen zu den Feldern finden sich direkt dahinter. Um sie zu kennzeichnen sind sie jeweils kursiv gesetzt und eingeklammert.

Abbildung 5: Mikrostruktur

In der Datenbank selbst werden die Lemmata mit ihren Phrasen und Erläuterungen zusammen gespeichert. Den Lemmata und Phrasen werden jeweils Übersetzungen zugeordnet, die wiederum Erläuterungen haben. Aus diesen Beziehungen ergibt sich der Artikel, wie er dem Benutzer angezeigt wird. Die genaue Organisation der Datenbank wird in Abschnitt 3.1 näher erklärt.

1.5.5 **Polysemie/Homonymie**

Da die Trennung dieser beiden Phänomene, sowie ihre Lexigraphische Behandlung nicht klar festgelegt sind[1], ist es in *inslex* dem jeweiligen Autor überlassen, wie er sie behandeln möchte. Die Datenbank- und Artikelstruktur ist so gestaltet, dass es möglich ist verschiedene Bedeutungen eines Wortes sowohl als eigene Artikel, als auch als innerhalb eines gemeinsamen Artikels anzulegen. Ein Entscheidungskriterium hierzu ist es, ob die bei einem Artikel angegebenen Phrasen für alle Begriffe des Artikels zutreffend sind. Bezieht sich ein Anwendungsbeispiel nur auf eine Bedeutung ist eine Aufteilung auf mehrere Artikel empfehlenswert.

[1] Siehe Abschnitt 1.1.2

Ein Lemma darf mit einer Lesart immer nur einmal in der Datenbank stehen. Es ist zulässig etwa *„aufnehmen“* mehrfach als Lemma zu vergeben, allerdings müssen in diesem Fall die Bedeutungen über das Erläuterungsfeld differenziert werden.

Nr	Hochdeutsch	Plattdeutsch
1	**aufnehmen** *(Arbeit aufnehmen)*	anfangen
2	**aufnehmen** *(Schulden / jdn in eine Organisation aufnehmen)*	opnehmen
3	**aufnehmen** *(Verhandlungen aufnehmen)*	sik an een Disch setten verhanneln
4	**aufnehmen** *(Fracht / Kapazitäten aufnehmen)*	stauen

Abbildung 6: Bedeutungen auf mehrere Artikel aufgeteilt

Nr	Hochdeutsch	Plattdeutsch
1	**ausrufen**	ansetten *(eine Trauerzeit)* utropen *(einen Katastrophenfall)*

Abbildung 7: Mehrere Bedeutungen in einem Artikel zusammengefasst

1.5.6 **Navigationskonzept**

Ein zentrales Anliegen des Navigationskonzeptes für die neue Benutzeroberfläche ist es die Navigation so einfach wie möglich zu halten. Grundsätzlich gliedert sich eine Suche in einer Datenbank in zwei Schritte: die Formulierung der Suchanfrage und das Anzeigen der Ergebnisse. Da es eventuell notwendig ist mehrere Suchanfragen zu formulieren, bis das gewünschte Ergebnis gefunden wird, ist es aus Usability-Gründen empfehlenswert den Wechsel zwischen Suche und Ergebnisanzeige so einfach und schnell wie möglich zu gestalten. Ideal ist es, wenn die Suchmöglichkeit auch während der Anzeige der Ergebnisse sichtbar bleibt (vgl. Hearst 2009, S.14).

Aus diesem Grund ist die Suche in *inslex* als eine einzige Seite gestaltet. Es besteht keine Notwendigkeit für den Benutzer zwischen Ergebnissen und

Suchmaske hin und her zu springen, alle Such- und Einstellmöglichkeiten sind jederzeit verfügbar.

Es ist zu erwarten, dass die Suche über das Suchfeld am häufigsten genutzt wird (vgl. Hearst 2009, S.1), aus diesem Grund ist sie prominent ganz oben links auf der Seite platziert. Die erweiterten Suchoptionen sind standardmäßig ausgeblendet. Für einen Großteil der Suchen werden die Voreinstellungen reichen, die erweiterten Optionen sind für komplexere Anfragen gedacht und würden ansonsten zu Verwirrung führen (vgl. Nielsen / Loranger 2006, S. 150).

Die alphabetische Liste folgt auf die Suchergebnisse. So ist sichergestellt, dass die Tabelle mit den Ergebnissen direkt nach einer Suche sichtbar ist und nicht erst zum Seitendende gescrollt werden muss. Benutzer welche die Liste nicht benötigen können so mit dem oberen Seitenbereich arbeiten und brauchen die Liste nicht weiter zu beachten.

Da der Fokus der Datenbank auf der Suche nach plattdeutschen Übersetzungen liegt, ist die Ergebnistabelle immer so gestaltet, dass die hochdeutschen Begriffe auf der linken Seite stehen und die Ergebnisse von Hochnach Plattdeutsch lesbar sind. Diese Aufteilung wird auch beibehalten, wenn eine hochdeutsche Übersetzung zu einem plattdeutschen Begriff gesucht wird. Ein Wechsel im Tabellenlayout würde den Nutzer verwirren, bei einer Suche in beiden Sprachen wären zudem zwei Tabellen mit unterschiedlicher Aufteilung notwendig.

Eine Bedienungsanleitung, sowie weitere Informationen zum Wörterbuch finden sich auf extra Seiten. Die Benutzung des Wörterbuches ist somit klar von der Bedienungsanleitung (den komplementären Wörterbuchteilen) getrennt.

Die oben genannten Prinzipien gelten auch für die Dateneingabe. Hier gilt, dass alle Änderungen am Datenbestand auf einer Seite vorzunehmen sind. Es ist möglich einem Datensatz in einem Arbeitsschritt etwas hinzuzufügen, einen Teil zu löschen und etwas Bestehendes zu verändern.

2 Beschreibung des fertigen Systems (Bedienungsanleitung)

2.1 Suchmöglichkeiten

Für den Benutzer gibt es mehrere Möglichkeiten die Datenbank nach Inhalten zu durchsuchen. Einerseits können über ein Eingabefeld Angaben gemacht werden, die daraufhin in der Datenbank gesucht werden, andererseits werden Teile der Datenbank als alphabetisch sortierte Liste ausgegeben und der Benutzer kann den gewünschten Eintrag direkt auswählen. Zusätzlich wird die Eingabe eines Suchbegriffes im Suchfeld durch automatisch generierte Suchvorschläge (Autocomplete oder Incremental Search genannt, vgl. Hearst 2009, S. 11) unterstützt. Über den Button *mehr Optionen* hat der Benutzer die Möglichkeit die Suche an seine Bedürfnisse anzupassen. Beim Laden der Seite sind diese Optionen ausgeblendet, um nicht sofort mit zusätzlichen Einstellmöglichkeiten zu verwirren. Für den Hauptteil der Benutzer sollten die Standardeinstellungen ausreichen. Bei jedem Laden der Seite werden diese wiederhergestellt und die zusätzlichen Optionen ausgeblendet, um zu verhindern, dass nach einem Neuladen der Seite noch (ausgeblendete) Optionen aktiviert sind, die zu unerwarteten Suchergebnissen führen können. Werden die zusätzlichen Suchoptionen nicht oder nicht mehr benötigt, können sie über den Button *weniger Optionen* ausgeblendet werden. Hierbei wird die Suche ebenfalls auf die Standardeinstellungen zurückgesetzt, damit keine nicht-sichtbaren Optionen angewählt bleiben. Der folgende Abschnitt stellt die einzelnen Suchoptionen vor und beschreibt den Umgang damit.

Abbildung 8: Seitenübersicht (Front-End)
1: Suchfeld
2: Radiobuttons zur Suchsprache
3: Checkboxen zu den durchsuchten Feldern
4: Ergebnistabelle
5: Tabs
6: Lemmaliste

2.1.1 Suche über das Suchfeld

Die Suche über das Suchfeld stellt die einfachste und komfortabelste Lösung dar die Datenbank zu durchsuchen. Diese Suchmöglichkeit ist auch die erste, die dem Benutzer, direkt oben links, präsentiert wird. Um über das Suchfeld nach einem Begriff zu suchen wird dieser eingegeben und per Klick auf *Suchen*, oder per Enter-Taste bestätigt.

Die Suche über das Suchfeld ist in der Standardeinstellung rechts trunkiert, es werden alle Begriffe gefunden, deren Anfang mit dem eingegeben Begriff übereinstimmt. Hierbei ist es unerheblich, ob der Suchbegriff lediglich Teil eines Wortes ist, oder der Anfang einer Wortkette. Eine Suche nach *„Schi“* findet so *„Schiff“*, eine Suche nach *„Geld“* findet *„Geld verdienen“*. Die Suche arbeitet unabhängig von der Groß- und Kleinschreibung. Es Spielt keine Rolle, ob nach *„warten“*, *„Warten“*, oder *„wARTen“* gesucht wird.

In der Standardeinstellung wird sowohl in der hochdeutschen, als auch in der plattdeutschen Sprache gesucht. Es werden lediglich die gespeicherten Lemmata durchsucht, nicht jedoch die Phrasen oder Erläuterungen. Werden mehrere Wörter in das Suchfeld eingegeben, so werden diese automatisch miteinander verknüpft, so dass nur Einträge gefunden werden, worin alle Wörter vorkommen. Hierbei ist die Reihenfolge der Suchwörter zu beachten, die der Reihenfolge im gesuchten Eintrag entsprechen muss.

2.1.2 **Eingrenzung der Suchsprache**

Über die Radio-Buttons zu Suchsprache lässt sich die Suchsprache einschränken. Sollte eine Suche über beide Sprachen zu viele nicht-relevante Treffer (Informationsballast) liefern, können diese so reduziert werden.

Es stehen drei Möglichkeiten zu Auswahl:
Hochdeutsch – Plattdeutsch durchsucht die plattdeutsche Sprache, es wird ein hochdeutscher Suchbegriff erwartet.
Plattdeutsch – Hochdeutsch durchsucht die hochdeutsche Sprache, es wird ein plattdeutscher Suchbegriff erwartet.
beide durchsucht sowohl die plattdeutsche, als auch die hochdeutsche Sprache. Die Sprache des Suchbegriffes spielt keine Rolle. Dies ist die Standardeinstellung.

2.1.3 **Präzisierung der durchsuchten Teile**

In der Standardeinstellung sind die durchsuchten Begriffe nur die hoch- und plattdeutschen Lemmata. Über die Checkbox *„auch Erläuterungen und Anwendungsbeispiele durchsuchen“* kann die Suche so erweitert werden, dass auch die Erläuterungen der Lemmata, sowie die Phrasen und deren Erläuterungen in die Suche einbezogen werden. Da es nicht zu erwarten ist, dass ein Benutzer weiß, was in diesem Zusammenhang mit dem Begriff *„Phrase“* gemeint ist, wurde für die Beschriftung der leichter verständliche (wenn auch teilweise unzutreffende) Begriff *„Anwendungsbeispiele“* gewählt. Ist diese Option aktiviert, gibt eine Suche nach *„See“* das Lemma *„warten“* als Treffer aus, da *„See“* in der Phrase *„Schiffe auf See warten lassen“* vorkommt, die zum Lemma *„warten“* gehört.

Über die Checkbox *nur ganze Wörter suchen* lässt sich die standardmäßig aktivierte Rechtstrunkierung ausschalten. Wird nur ein ganz bestimmter Begriff gesucht, lässt sich so der Informationsballast reduzieren. Ist der ge-

suchte Begriff Teil einer Wortkette wird er gefunden, ist er nur Teil eines Wortes zählt er hingegen nicht als Treffer. Eine Suche nach *„Stüern“* findet so den Begriff *„höhgere Stüern“*, nicht jedoch *„Stüerbescheed“*. Um den Anteil an nicht-relevanten Treffern weiter einzuschränken, unterscheidet diese Suche zusätzlich nach Groß- und Kleinschreibung, eine Suche nach *„Warten“* findet nicht *„warten“*.

Diese beiden Optionen lassen sich miteinander und mit der Einschränkung der Suchsprache kombinieren. So ist es z.B. möglich nur nach ganzen Wörtern im Plattdeutschen zu suchen.

2.1.4 **Suche über die Suchvorschläge**

Um den Benutzer bei der Eingabe von Suchbegriffen zu unterstützen unterbreitet das System Suchvorschläge. Diese, Autocomplete genannte, Funktion ist von zahlreichen Suchmaschinen, wie z.B. Google, bekannt. Wird mit der Eingabe eines Begriffes begonnen, öffnet sich unter dem Eingabefeld eine Liste mit Suchvorschlägen. Diese Vorschläge können mit der Maus, oder der Tastatur ausgewählt und per Enter-Taste, oder Mausklick bestätigt werden. Wird ein Vorschlag ausgewählt, wird er in das Suchfeld übernommen und eine Suche mit den momentan gewählten Einstellungen ausgeführt.

Abbildung 9: Liste mit Suchvorschlägen

Die Suchvorschläge basieren auf den gespeicherten Lemmata. Es wird automatisch eine rechtstrunkierte Suche nach den im Suchfeld stehenden Buchstaben ausgeführt und alle Treffer dieser Suche als Vorschlagsliste ausgegeben, für jeden zusätzlich eingegebenen Buchstaben wird eine neue Abfrage gestartet und die Liste entsprechend aktualisiert. Die Einstellungen für die Suchsprache werden hierbei berücksichtigt, ist als Suchrichtung z.B. *„Hochdeutsch – Plattdeutsch“* ausgewählt, werden nur plattdeutsche Suchvorschläge unterbreitet. Die Einstellungen der Checkboxen hingegen werden nicht übernommen. Erläuterungen und Phrasen in die Liste der Suchvorschläge mit einzubeziehen würde diese zu lang und damit unbenutzbar machen. Die Liste der Suchvorschläge nur nach exakten Treffern zu erstellen würde bedeuten, dass der gesamte Begriff eingegeben werden muss, bevor die Liste erscheint, in diesem Fall wäre sie nicht mehr notwendig. Es besteht zwar die Gefahr, dass der Benutzer keinen Treffer für eine Suche erwartet, da die Autocomplete Funktion nichts vorschlägt, allerdings sind die erweiterten Suchoptionen für erfahrene Benutzer gedacht. Es ist zu erwarten, dass sich das Verhalten der Autocomplete-Liste nach einer erfolgreichen Suche trotz fehlender Suchvorschläge erschließt.

2.1.5 **Suche über die Liste**

Am unteren Rand der Seite steht mit einer alphabetisch sortierten Liste eine weitere Suchmöglichkeit zur Verfügung. In dieser Liste sind alle hochdeutschen Lemmata in alphabetischer Reihenfolge aufgelistet. Da eine einzige Liste mit mehreren Tausend Einträgen nicht mehr zu überblicken ist, ist die Liste in die Buchstaben A bis Z unterteilt. So ist es möglich direkt zum gesuchten Anfangsbuchstaben zu springen. Ein Klick auf einen Begriff führt eine Suche nach dem gewählten Wort mit den Einstellungen „Hochdeutsch – Plattdeutsch“ und „nur ganze Wörter finden“ durch. Diese Einstellungen wurden gewählt, da davon auszugehen ist, dass der Nutzer bei einer Auswahl über die Liste gezielt zum jeweiligen Artikel springen möchte.

2.1.6 **Ausgabe der Ergebnisse**

Die Ergebnisse werden in einer Tabelle zwischen dem Eingabefeld und der Liste präsentiert. Das gesuchte Wort wird fett dargestellt, auch wenn es Teil eines Wortes ist, so wird dem Nutzer geholfen, einen Zusammenhang zwischen Sucheingabe und Ausgabe herzustellen (vgl. Hearst 2009, S. 8f).
Wird kein Treffer zu einem Suchwort gefunden, wird eine Meldung ausge-

geben. Zusätzlich wird die Benutzereingabe angezeigt, um auf eventuelle Eingabefehler aufmerksam zu machen (vgl. Nielsen / Loranger 2006, S.159).

2.2 Dateneingabe

Diese Seite stellt das Back-End des Systems dar. Hier können eingeloggte Benutzer die Datenbank verwalten. Es ist möglich Einträge zu ändern, zu ergänzen, zu löschen und neue Einträge zu erstellen. Alle Änderungen werden direkt in der Ausgabetabelle vorgenommen. Die Änderungen sind zunächst nur Vorschläge, erst bei einem Klick auf den Button *Speichern* wird die Datenbank verändert, der Vorgang kann jederzeit abgebrochen werden. Felder die nicht mit Text gefüllt sind, etwa ein Lemma ohne Erläuterung, werden bei der Trefferausgabe mit einem Platzhaltertext gefüllt (z. B. *„-keine Erläuterung-“*). Dieser Text soll lediglich darauf aufmerksam machen, dass an dieser Stelle eine Eingabe möglich ist. Er wird weder in der Datenbank gespeichert, noch bei der Suche über das Front-End mit ausgegeben. Der Fokus der Datenbank liegt auf der Übersetzung von hochdeutschen Begriffen ins Plattdeutsche. Sofern nicht anders erwähnt, bezeichnet beim Ändern eines Datensatzes *Lemma* den hochdeutschen Begriff und *Übersetzung* den dazugehörigen plattdeutschen Begriff. Für die Dateneingabe müssen JavaScript und Cookies aktiviert sein. Sie wurde für die Browser *Internet Explorer* und *Firefox* optimiert.

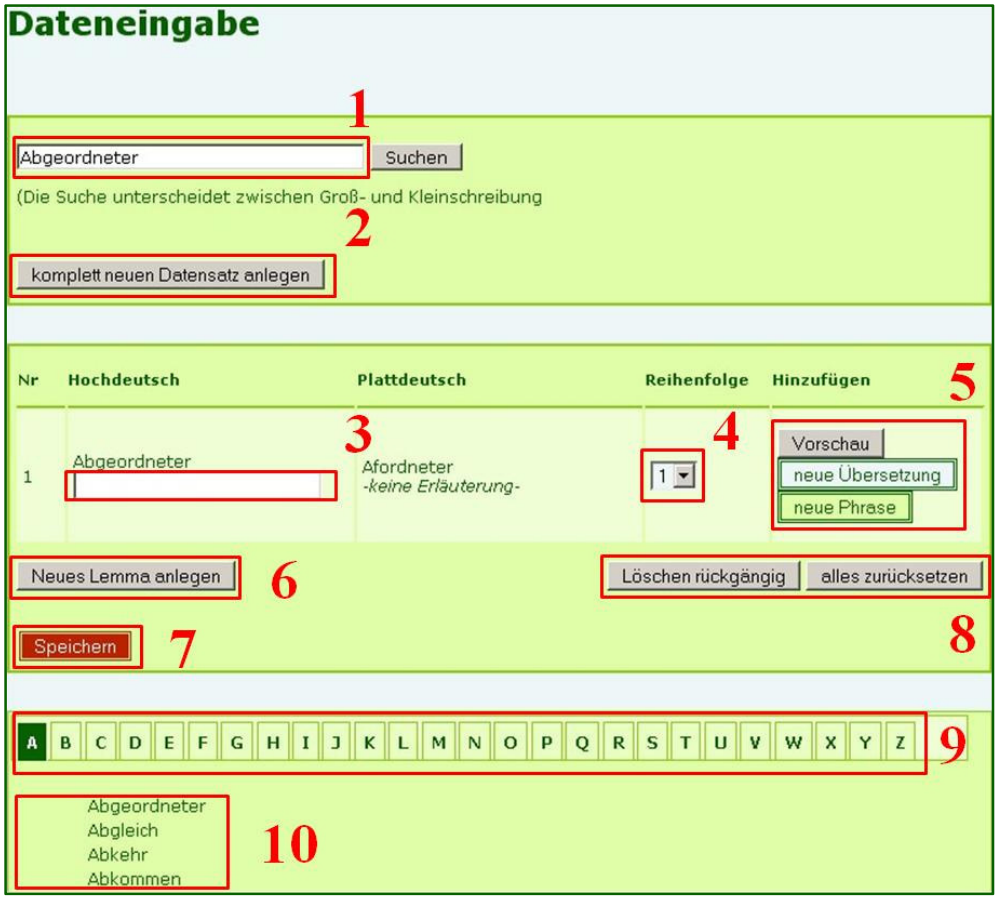


Abbildung 10: Dateneingabe
1: Suchfeld
2: Button für Neueingabe
3: Inputfeld (angeklickter Text)
4: Dropdown-Menü für Reihenfolge
5: Buttons für Vorschau und neue Inputfelder
6: Neues Lemma hinzufügen
7: Speichern-Button
8: Rückgängig-Buttons
9; Tabs
10: Lemmaliste

2.2.1 Suche nach Datensätzen

Der Änderungsmodus bietet die aus der Suche bekannten Suchmöglichkeiten. Im Unterschied zum Suchmodus funktioniert die Suche über das Suchfeld hier jedoch ohne Trunkierung, es werden nur die hochdeutschen Wörter und durchsucht und zwischen Groß- und Kleinschreibung unterschieden. Da bei einer Suche hier kein umfassender Überblick über den Datenbestand gegeben werden soll, sondern sehr gezielt nach einem bestimmten Datensatz

gesucht werden soll, erleichtert dies die Arbeit. Falls dennoch eine trunkierte Suche nötig ist, kann ein Prozentzeichen (%) als Platzhalter verwendet werden.

2.2.2 **Ändern von Datensätzen**

Ein angezeigter Datensatz kann geändert werden, indem der zu ändernde Begriff in der Ausgabetabelle angeklickt wird. Ein Klick öffnet ein Textfeld, welches mit dem Begriff gefüllt ist, dieser kann nun bearbeitet werden. Ein Klick außerhalb des Feldes oder ein Druck auf die Enter-Taste schließt das Textfeld und übernimmt den Text aus dem Textfeld in die Tabelle. Per Tab-Taste kann zum nächsten/vorherigen Begriff gesprungen werden. Wurde der angeklickte Text geändert, zeigt die rote Schriftfarbe des Textes die Änderung an. Wird die Änderung zu einem späteren Zeitpunkt rückgängig gemacht, wird auch die ursprüngliche Textfarbe wiederhergestellt.

Ein Datensatz kann ergänzt werden, indem neue Übersetzungen und/oder Phrasen hinzugefügt werden. Hierzu stehen die beiden Buttons *Übersetzung hinzufügen* und *Phrase hinzufügen* zur Verfügung. Beim Hinzufügen einer neuen Übersetzung wird ein neues Eingabefeld für eine Übersetzung und deren Erläuterung an das zum Button gehörende Lemma angefügt, sowie ein Button zum entfernen dieser Felder angezeigt. Diese Felder sind zunächst mit einem Platzhaltertext gefüllt. Es können beliebig viele Übersetzungen hinzugefügt werden. Wurden zu viele Eingabefelder hinzugefügt, kann mit einem Klick auf *-Übersetzung* das jeweils letzte Übersetzungsfeld wieder entfernt werden. Das Hinzufügen von neuen Eingabefeldern für die Phrasen funktioniert genauso. Da zu einer Phrase immer Begriffe in beiden Sprachen gehören werden hier immer Felder sowohl in der Spalte *Hochdeutsch* als auch in der Spalte *Plattdeutsch* erzeugt.

Über die Dropdownmenüs in der Spalte *Reihenfolge* kann die Reihenfolge in der die Übersetzungen und Phrasen angezeigt werden angepasst werden. Während die hochdeutschen Lemmata alphabetisch geordnet sind, sollen die Übersetzungen zu den einzelnen Lemmata nach semantischen Gesichtspunkten sortiert werden. Hier ist es denkbar, dass die gebräuchlichste Übersetzung am Anfang stehen soll, oder dass bestimmte Übersetzungen immer aufeinander folgen sollen. Die Menüs enthalten für jede eingegebene Übersetzung eine Zahl, die der Übersetzung zugeordnet werden kann. Je niedriger die zugeordnete Zahl ist, desto weiter oben in der Liste erscheint

die Übersetzung. Wird eine Zahl doppelt vergeben werden die Übersetzungen mit dieser Ordnungszahl alphabetisch angeordnet. So ist es etwa möglich eine bestimmte Übersetzung an den Anfang zu stellen, ohne sich Gedanken über die Sortierung der übrigen machen zu müssen.

Für jede neu hinzugefügte Übersetzung werden die Menüs entsprechend erweitert. Es müssen nicht alle Zahlen vergeben werden. Die konkreten Zahlen werden bei jedem Neuaufruf eines Eintrages neu vergeben, gemäß der aktuellen Reihenfolge der Einträge. Wurden für ein Lemma mit drei Übersetzungen die Zahlen 2-3-3 vergeben, so bleibt die gesetzte Reihenfolge bei einem erneuten Aufrufen erhalten. Die Zahlen lauten allerding 1-2-3. So ist sichergestellt, dass jederzeit jede gewünschte Anordnung der Übersetzungen eingestellt werden kann. Das Anpassen der Reihenfolge der Phrasen funktioniert ebenso.

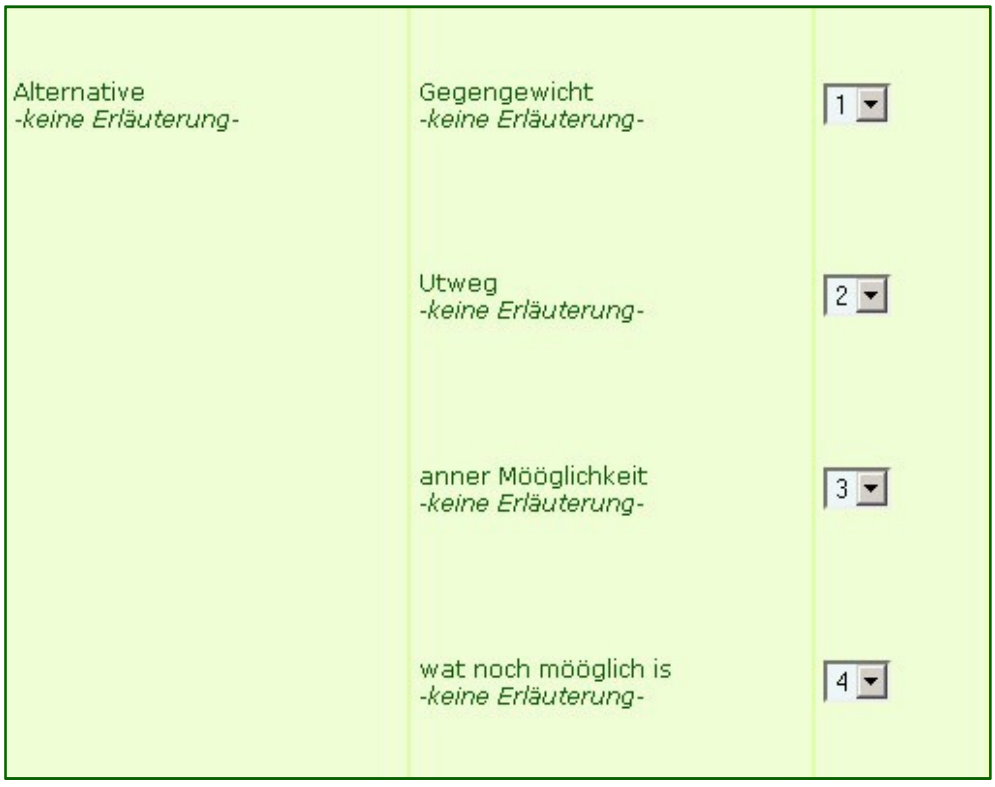


Abbildung 11: Reihenfolge – Originale Reihenfolge

Abbildung 12: Reihenfolge – geänderte Einstellungen

Abbildung 13: Reihenfolge – Eintrag nach dem Speichern und Neuaufrufen

Alle Änderungen für die gesamte Tabelle können über den Button *alles zurücksetzen* rückgängig gemacht werden. Um einen Eindruck des fertig bearbeiteten Datensatzes zu geben, kann über den Button *Vorschau* eine Vorschauansicht angezeigt werden. Hier erscheint das jeweilige Lemma so wie es dem Benutzer bei einer Suche angezeigt wird. Es werden keine Platzhaltertexte angezeigt und die Sortierung der Übersetzungen und Phrasen entspricht der der ausgewählten Reihenfolge.

2.2.3 **Neuanlegen von Datensätzen**

Ein komplett neuer Datensatz kann über den Button *komplett neuen Datensatz anlegen* erzeugt werden. Es werden Eingabefelder für ein Lemma und eine Übersetzung vorgegeben, weitere Felder können hinzugefügt werden. Ein Klick auf diesen Button erzeugt eine neue Ausgabetabelle, eventuelle Änderungen in aktuell angezeigten Begriffen gehen verloren. Aus diesem Grund bekommt der Benutzer eine Sicherheitsabfrage angezeigt, wenn in der Ausgabetabelle Änderungen vorgenommen wurden.

Soll einem Eintrag ein weiteres hochdeutsches Lemma hinzugefügt werden, oder mehrere Einträge hintereinander angelegt werden, ist dies über den Button *Neues Lemma anlegen* am Fuß der Ausgabetabelle möglich. In die-

sem Fall wird die Tabelle um eine Zeile für ein neues Lemma erweitert, ohne, dass bisherige Eingaben verloren gehen.

Bei der Eingabe von Erläuterungen ist zu beachten, dass die Klammern um die Erläuterung automatisch hinzugefügt werden. Diese Klammern sind nur Teil der Textformatierung für eine bessere Lesbarkeit und sind nicht als Teil der Erläuterung in der Datenbank gespeichert. Wird die Erläuterung bei der Eingabe in Klammern gesetzt, erscheint sie bei der Ausgabe doppelt geklammert.

2.2.4 **Löschen**

Einträge können aus der Datenbank gelöscht werden, indem sie angeklickt werden und der Text aus dem Eingabefeld entfernt wird. Nach dem Verlassen des Feldes wird der eben entfernte Text rot und durchgestrichen dargestellt. Dies zeigt an, dass der Text gelöscht werden soll. Der eigentliche Löschvorgang findet erst satt, wenn auf *Speichern* geklickt wird. Bis zu diesem Zeitpunkt kann jederzeit abgebrochen werden, ohne dass etwas aus der Datenbank gelöscht wird.

Wird ein Feld gelöscht, werden die davon abhängigen Felder ebenfalls gelöscht. Dies sind die dazugehörigen Erläuterungen, sowie bei Phrasen die Phrase in der jeweils anderen Sprache. Wird das Lemma gelöscht, wird der komplette Datensatz gelöscht, da alle Begriffe von diesem Lemma abhängen. Da dieser Vorgang alle neu angelegten Übersetzungen und Phrasen entfernt, existiert hier eine Sicherheitsabfrage um ein versehentliches Löschen zu verhindern.

Wurde ein Begriff geändert und daraufhin gelöscht, so wird der ursprüngliche Text durchgestrichen angezeigt, um zu kennzeichnen, dass dieser alte Text aus der Datenbank entfernt wird. Wird ein ursprünglich leeres Feld (enthält Platzhaltertext) mit Text gefüllt und dieser gelöscht, erscheint wieder der Platzhaltertext.

Ein als zu löschend markierter Begriff ist für Klicks gesperrt und kann nicht mehr geändert werden. Um die Löschmarkierungen für die gesamte Tabelle rückgängig zu machen gibt es den Button *Löschen rückgängig*.

2.2.5 **Prüfung der Eingaben**

Bevor die Nutzereingaben in der Datenbank gespeichert werden, wird geprüft, ob alle benötigten Felder ausgefüllt und ob doppelte Einträge vorhan-

den sind. Zu jedem Eintrag muss ein Lemma angegeben werden, die andern Felder sind optional. Es ist nicht möglich eine Übersetzung ohne eine dazugehöriges Lemma einzugeben. Wenn eine Erläuterung angegeben wird, muss dazu auch ein zu erläuternder Begriff angegeben werden. Zu jeder Phrase muss eine Übersetzung in der jeweils anderen Sprache vorhanden sein. Ist eine dieser Bedingungen nicht erfüllt ist es nicht möglich den Datensatz zu speichern. Um die Fehlersuche zu erleichtern wird die Tabellenzelle, in der eine Eingabe fehlt gelb gefärbt. Die Prüfung bezieht sich nicht auf Artikel, die komplett mit Platzhaltertexten gefüllt sind. Es ist möglich mehr als die tatsächlich benötigten Felder hinzuzufügen und diese einfach nicht mit auszufüllen.

Hochdeutsche Lemmata dürfen mit jeder Bedeutung nur einmal in der Datenbank gespeichert sein. Es ist möglich ein Lemma (z.B. *„aufnehmen“*) mehrfach mit unterschiedlichen Bedeutungen zu speichern (*„Arbeit aufnehmen“*, *„Fracht aufnehmen“*), eine Kombination aus Lemma und Erläuterung muss jedoch einmalig sein. Andernfalls wäre es für den Benutzer nicht erkennbar worin der Unterschied zwischen den Lemmata liegt und warum es zwei Artikel für ein Wort gibt. Um das Eingeben von doppelten Begriffen zu verhindern wird in zwei Schritten geprüft: Zuerst werden die eingegebenen Begriffe untereinander verglichen, um zu verhindern dass sie bereits bei der Dateneingabe doppelt eingegeben werden. Daraufhin, wird jedes eingegebene Lemma in der Datenbank nachgeschlagen, um zu verhindern, dass bereits gespeicherte Begriffe erneut gespeichert werden. Falls eine der beiden Prüfungen zutrifft, wird eine entsprechende Meldung ausgegeben, die die doppelten Begriffe benennt.

2.2.6 **Effekte der Dateneingabe**

Alle Begriffe, die in ein Lemma-Feld eingetragen werden, werden bei einer Suche in hochdeutscher Sprache durchsucht. Sie werden in der Liste angezeigt und bei der Erstellung der Suchvorschläge berücksichtigt.

Alle Begriffe, die in ein Übersetzungsfeld eingetragen werden, werden bei einer plattdeutschen Suche durchsucht. Sie werden bei der Erstellung der Suchvorschläge berücksichtigt. Wird bei einer Suche ein Treffer erzielt, werden immer alle Übersetzungen zu einem Lemma angezeigt.

In ein Phrasenfeld eingetragene Begriffe werden nur durchsucht, wenn dies vom Benutzer ausdrücklich eingestellt wird. Die Phrasen werden mit allen

darüber stehenden Begriffen (hoch- und plattdeutsch) verknüpft und bei einer Suche nach einem dieser Begriffe angezeigt.

Die Erläuterungen werden nur durchsucht, wenn der Benutzer dies ausdrücklich aktiviert. Sie werden immer mit angezeigt, wenn der übergeordnete Ausdruck in der Trefferliste auftaucht.

3 Technischer Teil

Dieser Teil der Dokumentation soll einen Überblick über die technischen Hintergründe von *inslex* liefern. Der Aufbau folgt den einzelnen „Schichten“ des Systems. Die Basis bildet eine MySQL Datenbank. Auf diese Datenbank wird über HTML-Formulare mittels PHP-Scripten zugegriffen. Die Benutzeroberfläche wird durch JavaScript und CSS-Dateien gestaltet. Jeder Abschnitt behandelt eine dieser Ebenen, es werden jeweils die übergeordneten Konzepte und Lösungswege anhand der einzelnen Programmfunktionen besprochen. Dieser Abschnitt soll den Code ergänzen und erklären, auch soll er eine Hilfestellung für eine Erweiterung oder Anpassung von *inslex* zu einem späteren Zeitpunkt bieten. Die Funktionen einzelner Details und Programmvariablen sind jeweils direkt im Quellcode dokumentiert. Hintergründe zur Vergabe von HTML-Klassen und IDs finden sich ebenfalls als Kommentare im Code. Bei der Programmierung lag das Ziel darin einen möglichst leicht verständlichen Code zu entwickeln, der auch im Nachhinein noch angepasst und verstanden werden kann. So wurden beispielsweise die Datenbankabfragen vorzugsweise auf mehrere Schritte verteilt, anstatt sie zu einer komplizierten Abfrage zusammenzufassen. Der PHP und jQuery Code wurde ebenfalls mit dem Ziel der Verständlichkeit geschrieben, die Geschwindigkeit der Programmausführung ist bei *inslex* sekundär.

Bei der Entwicklung kam ein Apache-Webserver (2.2.14, Win 32) mit MySQL (5.1.41) und PHP (5.3.1) zum Einsatz. Zusätzlich wurde die JavaScript-Bibliothek „jQuery“ (1.4.4) zusammen mit dem „UI Plug-in“ (1.8.9) verwendet. Die Benutzung von MySQL und PHP war durch die bestehende Webseite des INS vorgegeben, um eine reibungslose Einbindung zu gewährleisten.

Bei der Umsetzung der im ersten Teil entwickelten Struktur für das Wörterbuch gab es diese Ziele zu erfüllen:

- Die Struktur der Datenbank soll die Mikrostruktur der Artikel widerspiegeln.
- Die Inhalte der Datenbank sollen auf verschiedene Arten ausgegeben werden können. Einmal eingegebene Daten sollen nicht fest an die momentan bestehende Ausgabestruktur gebunden sein.
- Befehle sollen über eine benutzerfreundliche Oberfläche an die Datenbank geschickt werden können, ohne auf SQL-Kommandos zurückgreifen zu müssen.
- Die gespeicherten Daten sollen so vorliegen, dass es möglich ist das Wörterbuch um Funktionen zu erweitern oder die Daten in einem anderen Kontext wieder zu verwenden.
- Die Pflege des Datenbestandes soll möglich sein, ohne sich im Detail mit dem zugrundeliegenden Datenmodell beschäftigt zu haben.

Grundsätzlich wurde einer Lösung, die den Bedürfnissen des INS entspricht und eine einfache Benutzung des Wörterbuchs erlaubt Vorrang vor eventuellen lexikographischen oder datenbanktheoretischen Bedenken eingeräumt.

Die Abschnitte zu PHP und jQuery orientieren sich an der Dateistruktur des Quellcodes. Die jeweils besprochene Datei ist in Klammern hinter dem Abschnittnamen angegeben.

3.1 MySQL

3.1.1 Entitäten

Bei den zu speichernden Daten handelt es sich um die im bereits bestehenden *inslex*-System vorhandenen und die in Zukunft neu einzugebenden Einträge. Es handelt sich zum einen um Text, zum anderen um Informationen, die die Organisation der Daten betreffen, wie etwa Verknüpfungen zwischen Lemma und Phrase oder die Reihenfolge, in welcher Übersetzungen ausgegeben werden sollen. Die Daten sollen als Wörterbuchartikel gespeichert werden, ein Artikel setzt sich dabei aus den folgenden Entitäten und zusammen (die Attribute der Entitäten sind in Klammern dahinter aufgeführt:

- hochdeutsches Lemma (Lemma, Erläuterung)
- plattdeutsches Lemma (Lemma, Erläuterung)
- hochdeutsche Phrase (Phrase, Erläuterung)
- plattdeutsche Phrase (Phrase, Erläuterung)
- die Reihenfolge, in der Phrasen und Übersetzungen angezeigt werden sollen (Reihenfolge)

Zusätzlich enthält jede Instanz der Entitäten eine eigene Identifikationsnummer (Primärschlüssel). Es ist möglich, dass mehrere Instanzen einer Entität denselben Wortlaut haben. In diesem Fall handelt es sich trotzdem um zwei verschiedene Instanzen, da sie eine unterschiedliche Bedeutung haben, auch wenn sie gleich geschrieben werden. In der Datenbank werden die Begriffe Hoch- und Plattdeutsch jeweils mit *„h“* und *„p“* abgekürzt, so heißt z. B. die Tabelle der hochdeutschen Lemmata *„lemma_h“*.

3.1.2 **Tabellen und Verknüpfungen**

Ein Lemma kann jeweils über mehre Übersetzungen verfügen (Verknüpfungen zwischen hoch- und plattdeutschen Lemmata). Es kann über viele Phrasen verfügen und eine Phrase kann zu vielen Lemmata gehören. Zu einer Phrase sind ebenfalls viele Übersetzungen (Phrasen in der jeweils anderen Sprache) möglich. Zwischen den Entitäten wurden aus diesen Gründen *n-zu-m* Beziehungen gewählt. Die gewünschte Ausgabereihenfolge der Übersetzungen und Phrasen wird als Attribut an die jeweilige Verbindungsentität angehängt. Dies ermöglicht es, eine Übersetzung mehreren Lemmata an unterschiedlicher Position zuzuordnen.

Das fertige Entity-Relationship (ER)-Diagramm sieht wie folgt aus (Eine größere Darstellung findet sich im Anhang):

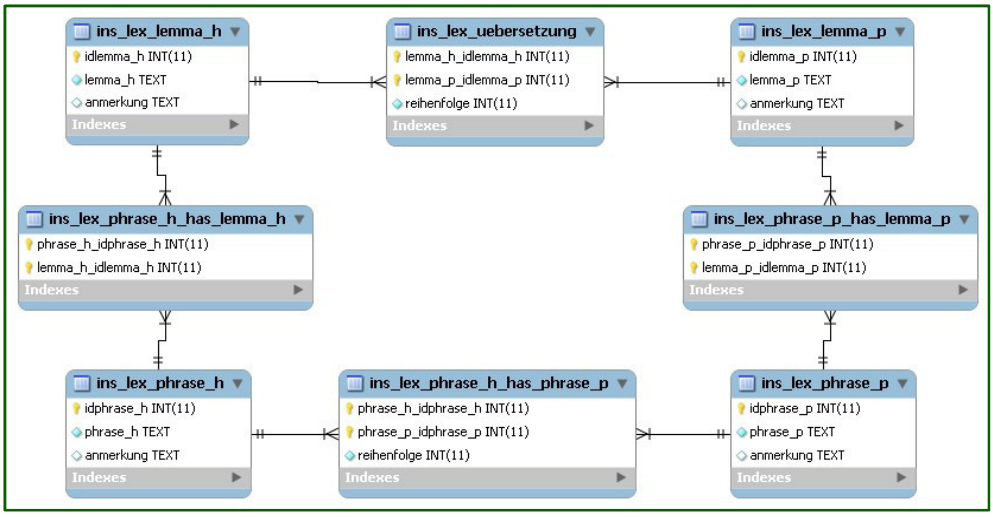


Abbildung 14: Entity-Relationship-Diagramm

Dieses Datenmodell ermöglicht zahlreiche Verknüpfungen. Im fertigen System werden nicht alle Möglichkeiten dieses Modells ausgereizt, die Einschränkungen kommen jedoch erst durch die Benutzeroberfläche zustande. So gibt das Datenmodell z. B. die Möglichkeit, ein plattdeutsches Lemma mit vielen hochdeutschen Lemmata zu verknüpfen, oder einer Phrase viele Phrasen der jeweils anderen Sprache zuzuordnen. In der Praxis ermöglicht die Benutzeroberfläche jedoch nur eine *1-zu-n* Beziehung zwischen hoch- und plattdeutschen Lemmata[1]. Wird ein plattdeutscher Begriff mehrfach eingegeben, wird er jedesmal als eine neue Instanz des *plattdeutschen Lemmas* angelegt und mit einem hochdeutschen Lemma verknüpft. Somit wird eine *n-zu-m* Beziehung durch die Eingabemaske unmöglich gemacht.

Ein Beispiel hierfür ist das plattdeutsche Lemma *„verhanneln“*, es lässt sich einerseits mit *„verhandeln“*, andererseits mit *„aufnehmen (Verhandlungen aufnehmen)“* übersetzen.

Nr	Hochdeutsch	Plattdeutsch
1	aufnehmen *(Verhandlungen aufnehmen)*	sik an een Disch setten **verhanneln**
2	verhandeln	**verhanneln** uthanneln

Abbildung 15: „verhanneln“

[1] Siehe Abschnitt 2.2

In der Datenbank wird *„verhanneln“* in diesem Fall zwei Mal gespeichert. Dies hat mehrere Vorteile. So ist es möglich, die Übersetzung von *„verhandeln“* zu verändern, ohne dabei die Übersetzung von *„aufnehmen“* zu verändern, da jeweils unterschiedliche Instanzen der Entität *lemma_p* geändert werden. Wäre dies nicht der Fall, müsste bei jeder Änderung unterschieden werden, ob die Instanz geändert werden soll, oder in mehrere neue Instanzen aufgespalten werden soll und welche Instanz auf welche Art verknüpft werden soll. Dies würde vom Benutzer eine Reihe von Entscheidungen verlangen, die nur zu treffen sind, wenn das Datenmodell bekannt ist, ohne einen dabei einen wirklichen Mehrwert zu bieten. Zusätzlich ist es so möglich den verschiedenen Instanzen unterschiedliche Erläuterungen zu geben.

Dasselbe Prinzip wird bei der Verknüpfung von Phrasen und Lemmata angewendet, hier kann ein Lemma mehrere Phrasen haben, eine Phrase kann über die Benutzeroberfläche jedoch immer nur einem Lemma zugeordnet sein.

Die Verknüpfung von Phrase zu Phrase ist im Datenmodell ebenfalls als *n-zu-m* Beziehung angelegt. In der Praxis wird jeder Phrase genau eine Phrase der jeweils anderen Sprache zugeordnet. Lässt sich eine Phrase auf unterschiedliche Arten übersetzen (was im Datenbestand eine Ausnahme darstellt), so wird sie als mehrere Instanzen behandelt. Die Phrase *„gewerbliche Sterbehilfe“* mit der Übersetzung *„Hölp-bi‘t-Starven* ist somit eine Instanz der Entität *phrase_h*, die Phrase mit der Übersetzung *„mit dat Starven Geld verdenen“* ist eine andere Instanz.

Nr	Hochdeutsch	Plattdeutsch
1	gewerblich	Geld verdienen mit
	gewerbliche Sterbehilfe	Hölp-bi't-Starven as Gewarv
	gewerbliche Sterbehilfe	mit dat Starven Geld verdenen

Abbildung 16: "gewerbliche Sterbehilfe"

Auch hier gilt, dass Änderungen so gezielt an einer Instanz möglich sind, ohne andere Artikel zu beeinflussen.

Die Verknüpfung zwischen plattdeutschen Phrasen und plattdeutschen Lemmata wird von der Benutzeroberfläche nicht angesprochen. Die Information, dass eine Phrase zu einem bestimmten Lemma gehört, ist jedoch

vorhanden und soll deshalb mit gespeichert werden, auch wenn die Zuordnung der plattdeutschen Phrasen zu den Lemmata letztendlich anders realisiert wird.[1]

Effektiv besteht die Datenbank aus *1-zu-1* und *1-zu-n* Beziehungen, trotzdem wurden im Datenbankmodell durchgängig *n-zu-m* Beziehungen gewählt. Dies hat den Grund, dass so eventuelle Änderungen am System leichter möglich sind und die *n-zu-m* Beziehungen nur einen leichten Overhead erzeugen. Eine Änderung am Datenbankmodell erfordert die Änderung bzw. komplette Neuentwicklung aller anderen Systembestandteile die darauf aufbauen, sowie eine eventuelle Neueingabe aller Daten und ist somit sehr aufwändig. Indem das Datenbankmodell mit *n-zu-m* Beziehungen arbeitet ist eine optimale Erweiterungsmöglichkeit gegeben. Sollte es etwa jemals erwünscht sein, eine Übersetzung zwei Lemmata zuzuordnen, oder eine Phrase mit mehreren anderen Phrasen zu verknüpfen, ohne sie als unterschiedliche Instanzen zu behandeln, ist dies möglich und wird von der Datenbank unterstützt. Die Änderungen sind lediglich an der Benutzeroberfläche vorzunehmen, indem beispielsweise neue Eingabefelder hinzugefügt werden. Die Basis des Datenbankmodells kann unverändert weitergenutzt werden und die enthaltenen Daten müssen nicht neu eingegeben werden. Zusätzlich bilden die Verbindungsentitäten einen Anknüpfpunkt für die Erweiterung um neue Funktionen. Auch bestehen die Entitäten so nur aus reinen Informationen, Meta-Informationen wie die Verknüpfung mit einer Übersetzung oder die Sortierreihenfolge sind in die Verbindungsentitäten ausgelagert.

3.1.3 **Datentypen**

MySQL bietet zahlreiche Datentypen für verschiedene Zwecke an. Diese unterscheiden sich sowohl im benötigten Speicherplatz, als auch darin wie sie die Geschwindigkeit der Datenbank beeinflussen. Die Datenbank ist mit 10.000 erwarteten Datensätzen[2] für MySQL leicht zu verwalten. Zudem bestehen die einzelnen Datensätze immer nur aus wenigen Wörtern. Somit sind die Unterschiede der Datentypen in der Verarbeitung hier nur gering. Aus diesem Grund wurde mit dem Typ *TEXT* für die Felder ein Datentyp

[1] Siehe Abschnitt 3.2.1
[2] Siehe Abschnitt 1.4.3

gewählt, der mit die geringsten Einschränkungen (z.B. bezüglich der Länge, Durchsuchbarkeit oder gültigem Zeichenformat) bietet.

Für die Schlüssel und die Reihenfolge wurde, wie üblich, *INT* als Datentyp gewählt, da so eindeutige Schlüssel vergeben werden können.
(vgl. Reese / Yarger / King 2003, S. 33ff)

3.1.4 **Normalisierung**

Zur Verhinderung von Lösch-, Einfüge- oder Änderungs-Anomalien soll die Datenbank normalisiert sein:

- Alle Attribute der Entitäten sind atomar, sie lassen sich nicht weiter zerlegen, es werden nicht mehrere Werte in einem Attribut gespeichert. Somit liegt die Datenbank in der ersten Normalform vor.
- Kein Attribut hängt nur von einem Teil der ID der Entität ab, somit liegt die Datenbank auch in der zweiten Normalform vor.
- Innerhalb der Entitäten hängt kein nicht-Schlüssel-Attribut von einem anderen ab, somit ist auch die dritte Normalform erfüllt.

Solange zudem jeder Schlüssel auf einen existierenden Wert verweist, ist die referenzielle Integrität der Daten gewährleistet. (vgl. Knorz 1997, S. 671ff ; Reese / Yarger / King 2003, S. 138ff)

In den Tabellen sind einige Begriffe mehrfach gespeichert, dies ist so beabsichtigt und stellt keine unnötige Redundanz der Daten dar[1]. In den Erläuterungsfeldern ist es theoretisch denkbar, dass mehrere Angaben gemacht werden und mit Trennzeichen gekennzeichnet werden. Auf eine weitere Aufteilung des Attributes *Erläuterung* wurde aus den in Abschnitt 1.5.4 genannten Gründen verzichtet.

3.1.5 **Speicher-Engine**

Bei MySQL Datenbanken stehen mehrere Speicher-Engines zur Auswahl. Die beiden gebräuchlichsten sind MyISAM und InnoDB. Die verschiedenen Engines unterscheiden sich in Funktionalität und Geschwindigkeit. Eventuelle Geschwindigkeitsunterschiede spielen bei einer Datenbank dieses Umfangs jedoch keine Rolle. Da InnoDB die aktuelle MySQL Standard-Engine darstellt, wurde sie für diese Datenbank gewählt. Zudem ermöglicht

[1] Siehe Abschnitt 3.1.2

es InnoDB, die referenzielle Integrität der Daten automatisch zu überwachen. Sollte es zukünftig erforderlich sein, dass mehrere Benutzer gleichzeitig Daten ändern möchten, stellt wieder InnoDB durch die Unterstützung von Transaktionen die bessere Wahl dar, zudmal auf die MyISAM Funktion der Volltextsuche verzichtet werden kann (vgl. MySQL InnoDB 1010 ; MySQL MyISAM 2011).

3.1.6 **Zeichensatz**

Als Zeichensatz für die Datenspeicherung stehen zahlreiche Möglichkeiten zur Auswahl. jQuery[1] kodiert die Daten für Ajax-Abfragen im Zeichensatz UTF-8. Da Konvertierungen von einem Zeichensatz zu einem anderen im Zweifelsfall zu Problemen führen, wurde UTF-8 auch als Kodierung innerhalb der Datenbank gewählt. MySQL bietet verschiedene Varianten des UTF-8 Zeichensatzes an. Hier wurde *UTF-8_bin* gewählt. In diesem Zeichensatz unterstützt MySQL das Speichern deutscher Umlaute und kann auch zwischen diesen unterscheiden. Allerdings unterscheidet dieser Zeichensatz zwischen Groß- und Kleinschreibung (case-sensitive) Ist dies nicht gewünscht, müssen die Abfragen entsprechend gestaltet werden[2].

Um eine durchgängige Verwendung von UTF-8 zu gewährleisten wird die Datenbankverbindung explizit auf UTF-8 gestellt. Die HTML-Ausgabe wird ebenfalls im Header auf UTF-8 gestellt, zusätzlich ist der Quellcode als UTF-8 kodierten Dateien gespeichert.

3.1.7 **Abfragen**

Hier soll auf die allgemeinen Besonderheiten der verwendeten Datenbankkommandos eingegangen werden. Die genauen Anfragen werden durch PHP-Scripte anhand der Benutzereingaben generiert.

Eine Suche hat das Ziel, einen oder mehrere Artikel auszugeben. Hierfür ist es notwendig alle zu einem Artikel gehörenden Daten zu finden. Der Ausgangspunkt für eine Suche ist immer die Tabelle *lemma_h*. Hierin werden alle Lemmata gesucht, die den Kriterien entsprechen und davon ausgehend die dazugehörenden Daten ermittelt. Wird als Suchbegriff kein hochdeutsches Lemma angegeben, sondern z. B. eine Übersetzung bei einer Suche nach einem plattdeutschen Begriff, werden im ersten Schritt alle zutreffen-

[1] Siehe Abschnitt 3.3.1
[2] Siehe Abschnitt 3.1.7

den Einträge der Tabelle *lemma_h* ermittelt und dann eine reguläre Suche ausgeführt. Eine Suche über mehrere Tabellen wird mittels des *JOIN*-Befehls realisiert. Die genaue Reihenfolge, in der die Tabellen durchsucht werden, hängt von der Art der Benutzereingabe und den Sucheinstellungen ab. Aus diesem Grund findet sich eine genaue Beschreibung des Suchvorgangs in Abschnitt 3.2.1.

Eine Beispielsuche nach dem Lemma *„Haus“*:

```
SELECT lh.idlemma_h AS idlemma_h, lh.lemma_h AS
lemma_h, lh.anmerkung AS erlaeuterung
FROM ins_lex_lemma_h lh
WHERE LOWER(lh.lemma_h) LIKE "haus%"
OR LOWER(lh.lemma_h) LIKE "% haus%")
ORDER BY LOWER(lh.lemma_h) COLLATE UTF8_general_ci
```

Da der verwendete Zeichensatz case-sensitive arbeitet (s.o.), werden alle Anfragen, wo dieses Verhalten nicht erwünscht ist, zuerst in Kleinbuchstaben umgewandelt und die Tabellenspalten durch den Befehl *LOWER* ebenfalls als Kleinbuchstaben behandelt. So ist gewährleistet, dass die Abfrage case-insensitive arbeitet. Um die Sortierung auch auf Umlaute zu beziehen, wird für die Sortierung der Zeichensatz *UTF8_general_ci* gewählt. In diesem wird nicht zwischen den Umlauten unterschieden, so dass z.B. *Ü*, wie gewünscht, unter *U* eingeordnet wird.

Die Parameter zur Einschränkung der Abfragen werden per *LIKE* Befehl übergeben. Je nach Sucheinstellungen werden die Platzhalterzeichen verschieden gesetzt. In diesem Beispiel sollen alle Einträge gefunden werden, die mit der Zeichenkette *„haus“* beginnen (bzw. genau *„haus“* lauten), oder vor *„haus“* noch weitere Zeichen haben, die durch ein Leerzeichen von *„haus“* getrennt sind.

Eine Verwendung von regulären Ausdrücken hätte sich hier angeboten, um sie Suchanfragen zu formulieren. Leider ist dies nicht möglich, da der MySQL Befehl *REGEXP* nicht auf Daten anwendbar ist, die in einem Zeichensatz Kodiert sind, der mehrere Bytes zum Speichern eines einzelnen Zeichens verwendet (er ist nicht *multibyte-safe).* Somit wird der verwendete Zeichensatz UTF-8 nicht unterstützt.

3.2 PHP

Der Zugriff auf die Datenbank erfolgt über HTML-Formulare. Die Benutzereingaben werden durch PHP in SQL-Queries eingebunden und abgeschickt, die Ausgabe der Ergebnisse erfolgt ebenfalls durch PHP. Die HTML-Formulare werden mittels einer PHP-Datei generiert, dies ist eine Voraussetzung für das Einbinden in die bestehende INS Webseite, beeinflusst jedoch nicht die Funktionen.
Sollen die PHP-Dateien modifiziert werden, ist darauf zu achten, sie UTF-8 kodiert abzuspeichern, da es sonst zu Problemen mit Vergleichsoperatoren kommen kann, die Umlaute enthalten (z. B. *if (Text == „-keine Erläuterung-„)*). Bei der UTF-8 Kodierung ist die Variante *„ohne Byte Order Mark“* (BOM) zu wählen. Andernfalls werden vor den eigentlichen Code einige den Zeichensatz betreffende Bytes eingefügt, die bereits als Ausgabe zählen und somit zu Problemen mit dem für Sessions benötigten PHP-Header führen.

3.2.1 **Suche (*suche_ausgabe.php*)**

Die Suche gliedert sich in drei Teile. Zuerst werden die IDs der auszugebenden Lemmata ermittelt; welche dies sind hängt von den Sucheinstellungen ab. Im nächsten Schritt werden die zu den gefundenen Lemmata gehörenden Übersetzungen und Phrasen ermittelt. Diese werden im dritten Schritt als Tabelle ausgegeben.

Die Abfragen an die Datenbank setzten sich aus einem statischen und einem dynamischen Teil zusammen. Der dynamische Teil besteht aus den Namen der zu durchsuchenden Attribute und der um Platzhalter erweiterten Nutzereingabe. Dieser Teil wird in einer Variablen gespeichert und die Variable in den statischen Teil eingefügt. Die Suche hat zwei verschiedene Sucheinstellungen, *trunkiert / nicht trunkiert*, sowie *alle Felder durchsuchen / nur die Lemmata durchsuchen*. Beide Einstellungen lassen sich kombinieren, so dass es insgesamt vier verschiedene Arten gibt die Abfragen zu erzeugen[1]. Der dynamische Teil der Suche erzeugt die Teile der Abfragen, die die Trunkierung betreffen. Je nach zu durchsuchenden Feldern werden diese in unterschiedliche Teile des statischen Teils eingefügt. Sofern nicht anders vermerkt, beziehen sich die folgenden Beispiele auf eine Suche in der Richtung *Hochdeutsch – Plattdeutsch*.

[1] Siehe Abschnitt 2.1

3.2.1.1 Dynamischer Abfrageteil

nicht trunkierte Suche: Soll die Suche nur ganze Wörter finden, gibt es vier Konstellationen, bei denen ein Treffer ausgegeben werden soll.

1. Es befindet sich nur der gesuchte Begriff im Feld
2. Auf den gesuchten Begriff folgen noch weitere Wörter
3. Vor dem gesuchten Begriff stehen weitere Wörter
4. Sowohl vor, als auch nach dem gesuchten Begriff stehen weitere Wörter

Diese vier Fälle werden mit *OR* verknüpft und in die Query eingefügt, eine Beispielabfrage findet sich in Abschnitt 3.1.7. Um nur einen Treffer auszugeben, wenn weitere Wörter folgen und nicht wenn der Suchbegriff Teil eines Wortes ist, werden die Platzhalterzeichen durch ein Leerzeichen vom Suchbegriff getrennt. Sollen wie im obigen Beispiel die Tabelle *lemma_h* nach *„Haus“* durchsucht werden, aber nur ganze Wörter gefunden werden, so lautet der Teil der Query nach dem *WHERE*:

```
… (lh.lemma_h LIKE "Haus"
OR lh.lemma_h LIKE "Haus %"
OR lh.lemma_h LIKE "% Haus"
OR lh.lemma_h LIKE "% Haus %") …
```

Da die nicht trunkierte Suche case-sensitive sein soll, wird hier auf den Befehl *LOWER* verzichtet.

trunkierte Suche: Bei einer trunkierten Suche gibt es fast dieselben Fälle, in denen ein Treffer ausgegeben werden soll, wie bei der nicht trunkierten Suche. Der Unterschied ist, dass die auf den Suchbegriff eventuell folgenden Zeichen nicht durch Leerzeichen vom Suchbegriff getrennt sein müssen. Da das Platzhalterzeichen % auch Leerzeichen mit einschließt, reichen zwei *LIKE* Befehle um den dynamischen Teil zu generieren. Der Teil der Query nach dem *WHERE* lautet:

```
… (LOWER (lh.lemma_h) LIKE "haus'%"
OR LOWER (lh.lemma_h) LIKE "% haus%")…
```

Da die trunkierte Suche die Groß- und Kleinschreibung ignorieren soll, wird der Suchbegriff durch die PHP-Funktion *mb_strtolower* in Kleinbuchstaben umgewandelt und der SQL-Befehl *LOWER* verwendet.

Insgesamt werden vier Variablen mit den benötigten *LIKE* und *WHERE* Statements erzeugt, je eine um die Lemmata, die Erläuterungen, die Phrasen

und die Erläuterungen der Phrasen zu durchsuchen. Im nächsten Schritt werden diese Variablen verwendet, um die kompletten Abfragen zu erzeugen.

Um das Finden von Einträgen zu erleichtern die aus mehreren Wörtern bestehen, werden alle Leerzeichen im Suchbegriff durch % ersetzt. Solange die Reihenfolge der Begriffe im Suchbegriff mit denen im gesuchten Eintrag übereinstimmt, wird ein Treffer erzielt, unabhängig davon, ob noch Zeichen dazwischen stehen (die einzelnen Suchbegriffe sind so mit einem UND verknüpft). Der Benutzer muss nicht alle Wörter des Begriffes eingeben. Eine Suche nach *„sik setten“* würde in *„sik%setten“* umgewandelt und so im Plattdeutschen den Begriff *„sik an een Disch setten“* finden. Um den Informationsballast zu reduzieren, spielt die Reihenfolge der Suchbegriffe weiterhin eine Rolle. Eine Suche nach zwei oder mehr Begriffen listet nur die Artikel auf, in denen beide Begriffe in dieser Reihenfolge vorkommen.

3.2.1.2 **Statischer Abfrageteil**

Dieser Teil enthält die zu durchsuchenden Tabellen, die oben beschriebenen Variablen mit den *WHERE* Klauseln werden hierin eingefügt. Die zu durchsuchenden Tabellen unterscheiden sich, je nachdem, ob nur die Lemmata oder auch die Erläuterungen und Phrasen durchsucht werden sollen.

Ist die Suchsprache Hochdeutsch und es sollen nur die Lemmata durchsucht werden, wird die Tabelle *lemma_h* auf Übereinstimmungen mit dem Suchbegriff durchsucht. Wird ein Treffer erzielt, wird die ID des Eintrages zur späteren Verwendung in einem Array gespeichert. Wird die Suche auf die Phrasen und Erläuterungen ausgedehnt, werden zwei andere Abfragen ausgeführt. Die erste Abfrage durchsucht ebenfalls die Tabelle *lemma_h,* mit dem Unterschied, dass zusätzlich die Erläuterungen durchsucht werden. Die zweite Abfrage durchsucht in der Tabelle *phrase_h* die Phrasen und die Erläuterungen. Ein *JOIN* mit der Tabelle *phrase_p* ermittelt die zur Phrase gehörende Übersetzung.

Bei einer Suche *Plattdeutsch-Hochdeutsch* wird die Tabelle *lemma_p* durchsucht und über ein *JOIN* mit *lemma_h,* das dazugehörige hochdeutsche Lemma ermittelt. Die ID dieses Lemmas wird im Ergebnisarray gespeichert. Für die Suche nach plattdeutschen Phrasen wird die Tabelle *phrase_p* über einen *JOIN* mit der Tabelle *phrase_h* verbunden und die hochdeutsche Übersetzung der Phrase ermittelt. Sodann wird über einen

weiteren *JOIN* das zu dieser hochdeutschen Phrase gehörende hochdeutsche Lemma ermittelt. Die Mittlertabelle zwischen *phrase_p* und *lemma_p* kommt hierbei nicht zum Einsatz. Dies hat den Grund, dass es plattdeutsche Phrasen gibt, denen kein Lemma zugeordnet ist.

Nr	Hochdeutsch	Plattdeutsch
1	**Ablösesumme**	
	10 Millionen Euro Ablösesumme	freeköpen för 10 Millionen Euro

Abbildung 17: Plattdeutsche Phrase ohne plattdeutsches Lemma

Würde die Verbindung zwischen plattdeutscher Phrase und hochdeutschem Lemma über das plattdeutsche Lemma hergestellt, könnte es nicht für solche Phrasen ohne Lemma funktionieren. Sofern vorhanden, soll eine direkte Beziehung zwischen den plattdeutschen Phrasen und Lemmata aber in der Datenbank gespeichert werden, da die Daten schließlich vorliegen, auch wenn sie momentan (noch) nicht genutzt werden[1].

Am Ende dieses Schrittes steht ein Array (*$lemma_array*), das die IDs der gefundenen Lemmata enthält. Anhand dieser IDs wird die Ausgabe generiert. Wurde bei einer Suche in beiden Sprachen gesucht, werden beide Arrays zu einem zusammengeführt. Um zu verhindern, dass Artikel, bei denen mehrere Felder die Suchkriterien erfüllen, doppelt ausgegeben werden, werden doppelte IDs aus den Arrays entfernt.

Das Script arbeitet im Folgenden mit diesem Array. Soll die Ausgabe geändert werden, steht somit eine klar definierte Ausgangsbasis zur Verfügung. Sollen die Suchanfragen geändert werden, ist dies ebenfalls problemlos möglich; solange das Ergebnis ein Array mit den IDs der Lemmata ist, muss an der restlichen Ausgabe nichts geändert werden.

[1] Siehe Abschnitt 3.1.2

Abbildung 18: Zusammensetzung einer Suchanfrage (Suchsprache Hochdeutsch, trunkierte Suche, keine Erläuterungen oder Phrasen durchsuchen)

3.2.1.3 **Artikelzusammenstellung**

Als nächstes werden die zu den IDs gehörenden Lemmata, Erläuterungen, Übersetzungen und Phrasen ermittelt, also alle Daten zusammengestellt, die für die Artikelausgabe notwendig sind. Es werden einige Daten erneut abgefragt, die bereits in den Abfragen des vorherigen Schrittes enthalten waren. Dies ist so beabsichtigt, um beide Schritte voneinander unabhängig zu halten.

Die Artikelzusammenstellung enthält drei Abfragen, je eine um die Lemmata, die Übersetzungen und die Phrasen und die jeweils dazugehörigen Erläuterungen abzufragen. Der Inhalt von *$lemma_array* wird hierbei Wert für Wert durchgegangen und die darin enthaltenen IDs als Basis für die Abfragen verwendet. Die Ergebnisse der Abfragen werden in fortlaufend nummerierten Arrays gespeichert, am Ende dieses Schritts gibt es drei Arrays mit Ergebnissen.

Da eine Abfrage für die Übersetzungen mehrere Ergebnisse liefern kann (wenn ein Lemma mehrere Übersetzungen hat), wird hier zum Speichern

Lemmata		Übersetzungen		
Index 1. Dimension	**Wert**	**Index 1. Dimension**	**Wert / Index 2. Dimension**	**Wert**
0	Steuer	0	0	Stüer
1	Steuerausfall	1	0	dat minner Geld bi de Stüern
			1	an Stüern kommt weniger rin
2	Steuererleichterung	2	0	de Staat will weniger an Stüern verlangen
			1	de Stüern daalsetten

Abbildung 19: Inhalt der Ergebnisarrays für Lemmata und Übersetzungen

auf ein mehrdimensionales Array zurückgegriffen. Die Nummerierung der ersten Dimension stimmt mit der Nummer des Ergebnisarrays der Lemmata überein, die zweite Dimension nummeriert die einzelnen Übersetzungen. Sollen drei Artikel ausgegeben werden, hat das Ergebnisarray der Lemmata drei Indizes (0, 1, 2), das Ergebnisarray der Übersetzungen ebenfalls. Um nun auf die zweite Übersetzung des zweiten Lemmas zuzugreifen, kann das mehrdimensionale Array der Übersetzungen am Index [1][1] angesprochen werden. Im weiteren Verlauf können so alle Übersetzungen für alle Lemmata ausgegeben werden, indem die beiden Arraydimensionen nacheinander durchgegangen werden. Für das Speichern der Phrasen wird dasselbe Prinzip angewendet.

Die Lemmata werden in alphabetischer Reihenfolge ausgegeben und bereits durch die Datenbankabfrage entsprechend sortiert. Die Übersetzungen und Phrasen werden gemäß der bei der Eingabe festgelegten Reihenfolge ausgegeben. Hierzu werden sie bei der Abfrage anhand des in der Vermittlungsentität gespeicherten Werts sortiert. Ist dieselbe Reihenfolge mehrfach vergeben, werden diese Einträge alphabetisch sortiert.

3.2.1.4 **Ausgabe der Ergebnisse**

Die gefundenen Ergebnisse sollen in einer Tabelle ausgegeben werden. Hierzu werden die Arrays mit den Ergebnissen in einer Schleife durchlaufen, die die einzelnen Lemmata durchgeht (die erste Dimension der Arrays). Innerhalb der Schleife läuft eine weitere Schleife, die die zweite Dimension der Arrays durchzählt und so die einzelnen Übersetzungen und Phrasen zu jedem Lemma ausgibt.

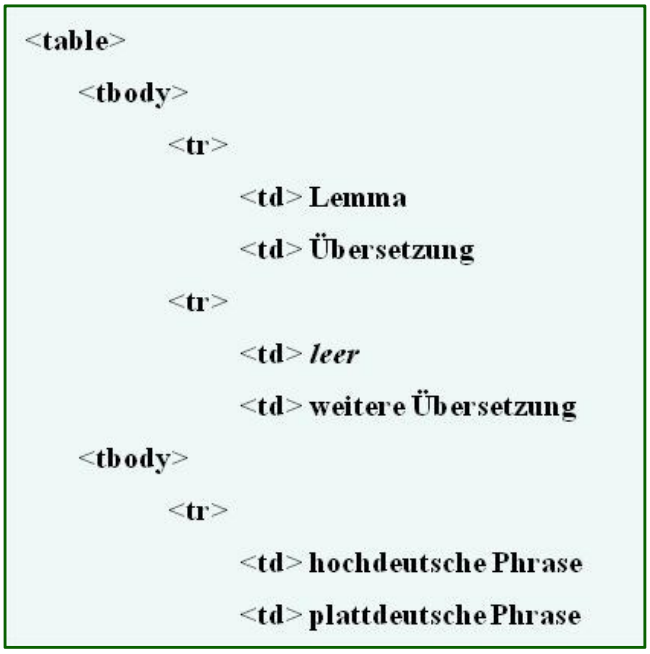


Abbildung 20: Schematische Übersicht der Ausgabetabelle

Der Arrayinhalt wird in den HTML-Code für Tabellenspalten und -zeilen eingebettet und ausgegeben. Das Lemma wird nur in der ersten Zeile eines Artikels ausgegeben. Die Erläuterungen werden, sofern vorhanden, in Klammern gesetzt. Das Lemma und die Übersetzungen stehen zusammen in einem *<tbody>,* die Phrasen in einem eigenen. Dies ermöglicht es alle Übersetzungen auf einmal, getrennt von den Phrasen, anzusprechen, z B. beim Hinzufügen/Entfernen von Extrafeldern.

3.2.1.5 **Autocomplete (*autocomplete.php*)**

Mit der Autocomplete-Funktion soll automatisch eine Abfrage nach den im Suchfeld stehenden Zeichen gestartet werden, ohne dass der Nutzer auf *Suchen* klicken muss. Die Ergebnisse sollen zur weiteren Verarbeitung in der JavaScript Object Notation (JSON) ausgegeben werden, der Nutzer sieht sie als eine Liste mit Suchvorschlägen[1].

Um die Liste mit den Suchvorschlägen zu erstellen, wird eine einfache Abfrage mit den eingegebenen Zeichen als (rechtstrunkiertem) Suchbegriff gestellt und das Ergebnis in einem Array gespeichert; es werden nur die Lemmata durchsucht. Für Hoch- und Plattdeutsch gibt es getrennte Abfragen, soll die Liste Vorschläge aus beiden Sprachen enthalten, werden die Arrays

[1] Siehe Abschnitt 3.3.3.6

zusammengeführt und doppelte Einträge entfernt. Sowohl der Suchbegriff, als auch die durchsuchten Spalten werden als Kleinbuchstaben behandelt, um die Suche case-insensitive zu gestalten. Die PHP-Funktion *json_encode* formatiert das Ergebnisarray als JSON, so dass es in jQuery weiter verwendet werden kann. Die Sortierung der Einträge wird in diesem Fall von jQuery übernommen.

3.2.1.6 **Liste (*liste.php*)**

Die alphabetische Lemmaliste besteht aus von A bis Z unterteilten Tabs. Jedes Tab ist ein Link, ein Klick darauf ruft die Datei *liste.php* auf, die eine Liste mit allen hochdeutschen Lemmata erstellt, die mit dem angeklickten Buchstaben beginnen. Das von *liste.php* gelieferte Ergebnis wird per Ajax als Liste unter den Tabs eingefügt.

Die Datenbankabfrage soll alle Lemmata finden, die mit einem bestimmten Buchstaben anfangen. Die Treffer sollen, in einen Link eingebettet, alphabetisch sortiert ausgegeben werden. Bei dieser Suche gibt es die Besonderheit, dass die Umlaute mit abgefragt werden sollen. Eine Suche nach allen Lemmata, die mit „A“ beginnen, soll auch alle finden, die mit „Ä“ beginnen. Hierzu wird die Abfrage so erweitert, dass bei einer Suche nach *A*, *O* oder *U* automatisch auch nach *Ä*, *Ö* oder *Ü* gesucht wird. Mit Umlauten beginnende Lemmata sollen so einsortiert werden, als würden sie nicht mit einem Umlaut beginnen. Dies wird erreicht, indem die Sortierung für diese Abfrage über den Zeichensatz *UTF8_general_ci* abgewickelt wird (`ORDER BY LOWER(lh.lemma_h) COLLATE utf8_general_ci`).

3.2.2 **Daten bearbeiten**

3.2.2.1 **Erzeugen der Tabelle (*aendern.php*)**

Die Basis für alle Datenänderungen ist eine durch die Datei *aendern.php* erzeugte Ausgabetabelle. Diese ähnelt der Ergebnistabelle bei der regulären Suche, zusätzlich zu den Suchergebnissen werden noch Inputmöglichkeiten zum bearbeiten der Daten angezeigt.

Die gefundenen Lemmata werden, von 0 ausgehend, durchnummeriert. Jedes Lemma steht (zusammen mit den Übersetzungen) in einem <tbody> mit der ID *„abschnitt + Nummer des Lemmas“*. Die Übersetzungen werden ebenfalls durchnummeriert und in Inputfeldern ausgegeben, deren Namen aus *„input_uebersetzung + Nummer des Lemmas + Nummer der Überset-*

zung“ bestehen. Für neue Übersetzungen lautet der erste Teil des Namens „*input_uebersetzung_neu“*. Durch die fortlaufende Nummerierung am Ende der Namen, können alle Inputfelder mittels Schleifen angesprochen werden. Es ist jederzeit ersichtlich, zu welchem Lemma eine Übersetzung gehört und ob das entsprechende Feld vom Nutzer neu hinzugefügt wurde. Die Phrasen und Erläuterungen folgen demselben Schema.

Das Erstellen der Abfragen und das Suchen nach Datensätzen ist ähnlich gelöst, wie bei der Suche, so werden auch hier die Ergebnisse in Arrays gespeichert und aus diesen Arrays mithilfe einer Schleife ausgelesen. Die Daten werden jedoch unterschiedlich ausgegeben. Die Ergebnisse der Abfragen werden zu späteren Verwendung in Session-Variablen gespeichert. Die Ausgabetabelle befindet sich in einem Formular, nach dem Abschicken stehen die Inhalte der Inputfelder als *Post*-Variablen zur Verfügung.

3.2.2.2 **Anzeige der Inhalte**

Die Artikelinhalte werden in jeder Zelle doppelt ausgegeben. Einmal als einfacher Text (in einem <*span*>-Tag) und einmal als Input-Feld. Der Nut-

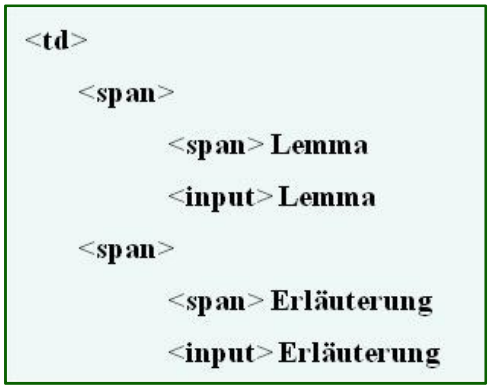


Abbildung 21: Schematische Übersicht einer Tabellenzelle

zer kann zwischen diesen beiden Anzeigen hin- und herschalten[1], es ist aber immer nur eine aktiv. Die Tabelle sieht so wesentlich übersichtlicher aus, als wenn sie nur aus Input-Feldern bestehen würde, zusätzlich ist sofort ersichtlich welcher Teil des Artikels gerade geändert wird.

[1] Siehe Abschnitt 3.3.3.2

Die Konstruktion aus Span- und Inputelement wird durch die Funktion *ausgabe* erzeugt, ihr werden die benötigten Daten als Parameter übergeben. Enthält ein Feld keinen Eintrag wird an diese Stelle ein Platzhaltertext eingefügt. Dieser besteht immer aus dem Wort *„keine“* gefolgt vom Feldtyp, links und rechts von einem Minuszeichen begrenzt (z.B. *„-keine Phrase-“*). So ist sichergestellt, dass der Benutzer sieht, dass an dieser Stelle nichts eingetragen ist und es gibt einen klickbaren Text, der es ermöglicht zum Inputfeld zu wechseln.

In der Spalte Reihenfolge befinden sich Dropdownmenüs, mithilfe derer die Reihenfolge der Ausgabe für Übersetzungen und Phrasen geändert werden kann. Diese werden über die PHP-Funktion *reihenfolge* erzeugt. Jedem Eintrag wird eine Zahl zugeordnet, je niedriger die Zahl, desto weiter oben steht der Eintrag in der Liste. Ist keine Übersetzung angegeben, wird trotzdem ein Dropdown mit dem Wert *1* erzeugt, um bei neu eingegebenen Übersetzungen auch der ersten einen Wert zuweisen zu können. Die Anzahl der Optionen im Dropdown richtet sich nach der Anzahl der Übersetzungen/Phrasen. Die aktuelle Position wird jeweils vorselektiert, um die Reihenfolge nicht bei jedem Aufruf des Artikels neu zuweisen zu müssen.

Am Ende der Ausgabe werden einige versteckte Textfelder erzeugt. Hier werden Daten gespeichert, die durch PHP erzeugt werden, aber in JavaScript benötigt werden. JavaScript kann den Inhalt dieser Felder auslesen. Da PHP Server- und JavaScript Clientseitig ausgeführt wird, benötigt ein Datenaustausch diesen Umweg.

3.2.2.3 **Übernehmen der Änderungen (*speichern.php*)**

In der Datei *speichern.php* werden die vorgenommen Änderungen in die Datenbank geschrieben. Das Formular in *aendern.php* übermittelt alle Inputfelder als *Post*-Variablen an *speichern.php*. Ziel ist es nun, geänderte Inhalte zu finden und sie an der richtigen Stelle in die Datenbank zu schreiben.

Die Suchergebnisse wurden als *Session*-Variablen gespeichert. Somit ist in *Session* der ursprüngliche Datenbankinhalt gespeichert, in *Post* hingegen der Inhalt, der eventuell vom Nutzer geändert wurde. Die Inhalte dieser Variablen werden abgeglichen, stimmen sie nicht überein, liegt eine Änderung vor.

Session und *Post* legen die Inhalte jeweils in Arrays ab, die Indexnummern stimmen bei beiden überein. Das von der Suche zuerst ausgegebene Lemma ist beispielsweise unter der Indexnummer **0** zu finden (*$_SESSION['lemma_array'][**0**]*). Es wird in einem Inputfeld mit dem Namen *input_lemma_**0*** gespeichert und ist nach dem Abschicken unter *$_POST [input_lemma_**0**]* verfügbar. Indem diese Indexnummern in Schleifen durchgegangen werden, ist es möglich, bei allen Begriffen den ursprünglichen mit dem aktuellen Wert zu vergleichen.

Stimmt der Inhalt von *Session* an einer bestimmten Indexnummer nicht mit dem von *Post* an derselben Indexnummer überein und ist der Inhalt des Inputfelds (*Post*) weder Platzhaltertext noch leer, so wird die entsprechende Funktion zum Ändern des Feldes aufgerufen. Diese Funktion sendet einen Update-Befehl an die Datenbank, das zu ändernde Feld wird über die in *Session* angegebenen IDs festgelegt, der einzufügende Text über den in *Post* gespeicherten Inhalt des Inputfelds.

3.2.2.4 **Neueingabe (*speichern.php*)**

Zusammen mit eventuell geänderten Daten werden im *Post*-Array auch neu eingegebene Begriffe gespeichert. Da sich die Queries zum Ändern von denen zur Neueingabe unterscheiden, ist es notwendig die neu eingegebenen Begriffe von den bereits bestehenden zu trennen und gesondert zu behandeln. Dies wird durch zwei Zählvariablen gelöst. Die erste zählt die Anzahl der Ergebnisse der Datenbankabfrage, die zweite die Anzahl der in *Post* gespeicherten Datensätze. Aus der Differenz lässt sich auf die Indexnummern der neu eingefügten Begriffe schließen, da diese immer am Ende des Arrays stehen (die Eingabefelder für neue Begriffe werden am Schluss der Tabelle eingefügt). Gibt es beispielsweise 4 Datensätze im *Post*-Array, sind die Indexnummern 0 bis 3 vergeben. Hat die ursprüngliche Abfrage drei Ergebnisse geliefert, sind unter den Nummern 0 bis 2 alte Begriffe gespeichert (die auf Änderungen geprüft werden), die Indexnummer 3 enthält einen neuen Datensatz.

Die Insert-Queries fügen als Primärschlüssel den Wert *„NULL“* ein, da die Schlüsselfelder auf Auto-Increment gesetzt sind, ist so die Einzigartigkeit der IDs gewährleistet. Enthält ein Eingabefeld einen Platzhaltertext, wird ebenfalls *„NULL“* eingetragen, da dieser Platzhalter nicht in der Datenbank gespeichert werden soll.

Abbildung 22: Schematische Übersicht der Schleifen in speichern.php - Es wird geprüft, ob die jeweiligen Elemente vorhanden sind. Ist dies der Fall wird mit den weiter eingerückten Prüfungen fortgefahren.

Die Schleifenstruktur von *speichern.php* sorgt gleichzeitig für eine Prüfung der Daten. Versagt die JavaScript-Prüfroutine[1] und es kommt beispielsweise ein Datensatz ohne Lemma an, so wird bereits die erste Schleife nicht ausgeführt und die restlichen Daten nicht in die Datenbank geschrieben.

Beim Ändern von Daten wird immer nur ein sehr kleiner Ausschnitt der Datenbank angezeigt und damit zur Änderung bereitgestellt. Es wäre also möglich gewesen auf die Prüfung zu verzichten, ob ein Begriff auch tatsächlich geändert wurde und alle angezeigten Begriffe, geändert oder nicht, in die Datenbank zu schreiben. Die momentane Lösung ist deutlich aufwändiger, bietet aber Vorteile und Erweiterungsmöglichkeiten. So ist es z. B. möglich

[1] Siehe Abschnitt 3.3.3.5

die Datenbank um Abfragen zum Änderungsdatum zu erweitern, oder alle Änderungen, die in einem bestimmten Zeitraum erfolgt sind, rückgängig zu machen.

3.2.2.5 **Löschen (*speichern.php*)**

Wenn die Inhalte auf Änderungen überprüft werden, wird die Datenbank nur aktualisiert, sofern das jeweilige Inputfeld nicht leer ist. Ist es hingegen leer, ist dies das Signal die entsprechenden Daten zu löschen. Hierbei ist darauf zu achten, so zu löschen, dass die Integrität der Datenbank bestehen bleibt.

Soll eine Erläuterung zu einem Begriff gelöscht werden, wird diese durch einen Update-Befehl auf den Wert *„NULL“* gesetzt. Wird der übergeordnete Begriff gelöscht, wird die ganze Instanz der jeweiligen Entität durch einen Delete-Befehl gelöscht. In diesem Fall werden auch die Einträge in den Vermittlungsentitäten gelöscht. InnoDB sorgt zwar automatisch dafür, dass die Fremdschlüssel mit gelöscht werden, das manuelle Löschen bedeutet jedoch nur einen geringen Mehraufwand und sorgt dafür, dass das Script unabhängig von der gewählten Speicher-Engine funktioniert.

3.2.2.6 **Vorschau (*vorschau.php*)**

Die Datei *vorschau.php* soll eine Tabelle erzeugen, wie sie auch von der regulären Suchausgabe erzeugt wird. Allerdings sollen hierin nicht nur die Daten aus der Datenbank, sondern auch alle neu eingefügten oder geänderten Begriffe enthalten sein. Zusätzlich soll die Reihenfolge der Begriffe mit der in der Änderungstabelle gewählten Reihenfolge übereinstimmen. Die Datenbasis ist, im Gegensatz zur normalen Suche, keine Datenbankabfrage, sondern der Inhalt der Inputfelder. Die fertige Tabelle soll als Popupfenster ausgegeben werden, für jedes Lemma wird eine eigene Tabelle erzeugt[1]. Der Inhalt der Textfelder wird als Post-Array an *vorschau.php* übergeben. Diese Inhalte (Übersetzung, Erläuterung und Reihenfolge) werden für jede Übersetzung in ein zweidimensionales Array geschrieben. Die erste Dimension dieses Arrays wird sodann anhand der Werte für *Reihenfolge* (aus der zweiten Arraydimension) sortiert. Somit stehen alle Übersetzungen zu einem Lemma in der vom Nutzer gewählten Reihenfolge im Array und kön-

[1] Siehe Abschnitt 3.3.3.9

nen durch eine Schleife als Tabelle ausgegeben werden. Mit den Phrasen wird genauso verfahren.

3.3 jQuery

3.3.1 Einführung

Die Grundfunktionalitäten von *inslex* sind durch die PHP Teile gegeben. Da PHP auf dem Webserver läuft, sind damit jedoch nur statische Webseiten möglich, es wird immer eine ganze Seite an den Client geschickt. Sollen nur Teile einer Seite verändert werden (z. B. wenn auf einen Button geklickt wird), erfordert dies das erneute Laden der gesamten Seite. Dies nimmt e Zeit in Anspruch, zudem gehen bereits getätigte Eingaben verloren.

Inslex kann gleich an mehreren Stellen von einer Dynamik in der Webseite profitieren. Beispielsweise sollen die Suchergebnisse direkt nach der Suche angezeigt werden, ohne die Suchmaske erneut laden zu müssen und es soll dem Nutzer ermöglicht werden, beliebig viele neue Felder für die Dateneingabe hinzuzufügen, ohne die Inhalte der bereits bestehenden Felder zu verlieren. Dynamische Webseiten lassen sich erreichen, in dem der entsprechende Code mittels JavaScript direkt beim Client ausgeführt wird. Bei der Gestaltung der dynamischen Seitenelemente kam die JavaScript-Bibliothek jQuery zum Einsatz, um das Arbeiten mit JavaScript zu erleichtern.

„jQuery ist eine Open Source JavaScript-Bibliothek, die die Interaktionen zwischen einem HTML-Dokument oder genauer dem Document Object model (DOM) und JavaScript vereinfacht. […] Insbesondere vereinfacht jQuery das Durchlaufen und Bearbeiten des HTML-Dokuments, den Umgang mit Browser-Events, DOM-Animationen, Ajax-Interaktionen und JavaScript-Entwicklung für verschiedene Browser.“
(Lidley 2010, S. 1f)

jQuery wird als JavaScript Datei in das Dokument eingebunden. Anstatt direkt JavaScript-Befehle zu schreiben, wird die jQuery Bibliothek über eine eigene Syntax angesprochen und daraus das eigentliche JavaScript generiert. Der ausschlaggebende Punkt für die Verwendung dieser Bibliothek war, dass sie eine browserunabhängige Entwicklung ermöglicht. Anstatt den verwendeten Browser abzufragen und den Code per Hand an die Eigenarten jedes Browsers anzupassen, kann ein Code für alle geschrieben werden.

jQuery ermöglicht, dass der JavaScript-Code unter allen gängigen Browsern funktioniert. jQuery erleichtert die Manipulation des DOM und die Ausführung von Ajax-Abfragen ungemein, beides wird für *inslex* benötigt. Darüber hinaus ist jQuery gut dokumentiert und es existiert eine breite Nutzerbasis. Die Funktionen von jQuery können durch Plug-Ins zusätzlich erweitert werden. Bei der Arbeit an *inslex* kam das Plug-in *„jQuery UI“*(User Interface) zum Einsatz. Die Plug-ins werden als JavaScript-Datei eingebunden und stellen zusätzliche Funktionen bereit.

Die jQuery Befehle werden in ein <*script*> Tag im HTLM-Code eingebettet. Es ist möglich zwischen normalem JavaScript und der jQuery-Syntax zu wechseln. Die jQuery Funktion wird mit einem Dollarzeichen ($) gestartet (alternativ kann auch *„jQuery“* verwendet werden). jQuery arbeitet mit drei grundlegenden Konzepten: *Finden*, *Verketten* und die *implizite Iteration* (vgl. Lindley 2010, S. 4ff). Nähere Informationen zu den im konkreten Fall verwendeten Methoden finden sich als Kommentare im Quellcode. Sofern nicht anders angegeben, stammen die Informationen zu den einzelnen jQuery Befehlen aus der offiziellen Dokumentation (jQuery Dokumentation 2011). Informationen zum jQuery UI Plug-in finden sich unter (jQuery UI 2011).

Finden: Die Basisfunktionalität von jQuery besteht darin ein (oder mehrere) Elemente im DOM zu finden und diese zu bearbeiten. Zum Suchen werden *Selektoren* verwendet. Die Selektoren orientieren sich an der Schreibweise der Cascading Style Sheets (CSS) und werden in runde Klammern gesetzt, jQuery durchläuft den DOM und findet alle Elemente, auf die der jeweilige Selektor zutrifft. Die am häufigsten verwendeten Selektoren suchen nach einem bestimmten Tag, einer Klasse, oder einer ID. Um beispielsweise alle <*div*> Elemente in einer Seite zu finden wird zuerst die jQuery Funktion mit einem $ aufgerufen und der Selektor für ein Tag in runden Klammern dahinter gesetzt, dieser besteht einfach aus dem Namen des Tags: `$(“div“)`. Um eine Klasse zu finden wird dem Klassennamen ein Punkt vorangestellt, um eine ID zu finden eine Raute.
`$(".gelb")` findet alle Elemente mit der Klasse *„gelb“*,
`$("#123")` findet das Element mit der ID *„123“*
Die Selektoren lassen sich miteinander kombinieren: `$("td.gelb")` findet alle <*td*>, die die Klasse *„gelb“* haben.

Die so gefundenen Elemente können bearbeitet werden. Hierzu wird die entsprechende Methode, durch einen Punkt getrennt, angehängt. Die Methode *text()* ändert den Textinhalt eines Elements, der neue Text wird als Parameter in die Klammern geschrieben: *text("neuer Text").* Um das Element <*div id="123"*> mit dem Text *„Hello World"* zu füllen lautet das Kommando: `$("#123").text("Hello World").`

Verketten: Einzelne jQuery-Methoden lassen sich miteinander verketten. Jede Funktion gibt die ausgewählten Elemente wieder zurück, so dass es möglich ist mehrere Befehle auf eine Auswahl anzuwenden, oder eine Auswahl erneut zu filtern. Um das Element <*div id="123"*> zuerst, wie oben beschrieben, mit Text zu füllen und ihm dann die Klasse *„gelb"* zu geben, können die beiden folgenden Kommandos verwendet werden:
`$("#123").text("Hello World")` und
`$("#123").addClass("gelb").`
Insbesondere bei umfangreichen Selektionen und Manipulationen ist es empfehlenswert die Verkettung zu nutzen. Die einzelnen Befehle werden, durch einen Punkt getrennt, hintereinander geschrieben:
`$("#123").text("Hello World").addClass("gelb")`

Implizite Iteration: jQuery verwaltet alle durch eine Selektion gefundenen Elemente als ein sog. *Wrapper-Set*. Die Elemente dieses Sets werden (von den meisten Methoden) automatisch durchlaufen, ohne dass es notwendig ist eine explizite Schleife anzugeben. Wenn eine Seite mehrere <*div*> Elemente hat und das Kommando `$("div").text("Hello World")` ausgeführt wird, werden alle <*div*> Elemente der Seite in das Set aufgenommen. Die folgende Methode (*text*) wird somit auf alle diese Elemente angewandt und es werden alle <*div*> mit dem entsprechendem Text gefüllt.

3.3.2 Häufig verwendete Befehle

Einige Befehle und Programmkonstruktionen kommen bei diesem Projekt immer wieder vor und sollen deshalb näher erläutert werden.

Anonyme Funktionen: Ist es erforderlich mehrere Methoden zu einem Schritt zusammenzufassen und auszuführen bevor in der Verkettung der Methoden weitergegangen wird, können sie zu einer Funktion zusammengefasst werden. Diese Funktionen stehen nur für die aktuelle Auswahl zur

Verfügung und bekommen keinen Namen zugewiesen. Sie werden durch das Wort *„function()“* eingeleitet, der jeweilige Code folgt in spitzen Klammern dahinter. (Siehe auch *.each()* im folgenden Abschnitt)

.each(): Einige Methoden beeinflussen nur das erste Element eines Sets, die implizite Iteration (s.o.) greift hier nicht. In diesen Fällen kann mit der Methode *.each()* explizit über alle Elemente des Sets gegangen werden. Der für die einzelnen Elemente auszuführende Code wird als *anonyme Funktion* als Parameter übergeben.

Beispiel:
Die Methode *.attr()* wird genutzt, um ein bestimmtes Attribut eines Tags auf einen bestimmten Wert zu setzen. Sie beeinflusst jedoch immer nur das erste Element einer Auswahl. Um sie auf alle Elemente der Auswahl anzuwenden, wird auf die Methode *.each()* und eine anonyme Funktion zurückgegriffen.

```
$("input").each (
   function(){
     $(this).attr('tabindex',i);
     i++;
  }
);
```

(Für jedes Element vom Typ <*input*> wird eine Funktion aufgerufen, die das Attribut *„tabindex“* auf den Wert der Variable *i* setzt und *i* um eins erhöht.)

Ein weiterer häufiger Anwendungsfall ist es, wenn auf ein gefundenes Element erst weitere Selektoren und danach Methoden angewendet werden sollen, etwa wenn eine Menge an Elementen ermittelt wird, der Code aber immer auf die jeweiligen Geschwisterelemente bezogen werden soll.

Event-Handler: Es existieren verschiedene Möglichkeiten, Ereignisse (Events), wie das Klicken eines Buttons oder einen bestimmten Tastendruck, mit jQuery zu managen. Hier kam die Methode *.live()* zum Einsatz. Diese knüpft ein Event an alle bestehenden Elemente eines bestimmen Typs an. Werden neue Elemente zur Seite hinzugefügt (z. B. neue Felder zur Dateneingabe eingeblendet) erkennt *.live()* diese und verknüpft das Event auch mit den neuen Elementen. Hierin unterscheidet sich diese Methode von anderen, wie z. B. *.bind()*.

Beim Umgang mit Event-Handlern ist darauf zu achten, dass der Verknüpfungsvorgang zwischen Event und auszuführender Funktion nur ein Mal ausgeführt wird. Andernfalls wird die jeweilige Aktion entsprechend mehrfach ausgeführt.

Code beim Laden ausführen: Bestimmte Kommandos sollen automatisch beim Laden der Seite ausgeführt werden. Unter jQuery steht hier der Befehl *$(document).ready()* zur Verfügung. Der in *.ready()* enthaltene Code wird ausgeführt, sobald das DOM fertig erstellt ist, aber noch bevor der komplette Seiteninhalt geladen ist. So kann JavaScript möglichst schnell ausgeführt werden, ohne dass auf das Laden von Bildern oder externen Quellen gewartet werden muss.

.find(): Die Methode *.find()* durchsucht alle Kindelemente eines Elements. Mit `$("#123").find(".gelb")` werden alle Elemente mit der Klasse *„.gelb“* gefunden, aber nur wenn sie in einem Element mit der ID *„123“* enthalten sind.

3.3.3 Die Funktionen im Detail

Der folgende Abschnitt soll besondere Aspekte und Lösungswege bei der Arbeit mit jQuery erläutern. Details zu den verwendeten Befehlen und Variablen finden sich als Kommentar im Code.

3.3.3.1 Ergebnisanzeige

Inslex verwendet an mehreren Stellen Ajax-Abfragen (Asynchronous JavaScript and XML). Die Tabelle mit den Suchergebnissen, die Autocomplete- Funktion und die Liste der Lemmas werden per Ajax in die bestehende Seite eingefügt.
Für die Ergebnisanzeige wird die Methode *.load()* verwendet. In ihr wird die zu ladende Datei und ein Zielelement für die Ausgabe genannt. An die aufgerufene URL (*suche_ausgabe.php*) wird der Inhalt des Suchfeldes und die gewählten Suchoptionen (entsprechend formatiert) angehängt.

```
$("#ausgabe").load("suche_ausgabe.php
?Suchwort=Abgeordneter
&sprache=hochdeutsch
&trunkierung=ganzes_wort");
```

jQuery interpretiert die URL mit angehängten Daten automatisch als GET-Abfrage. Die Ausgabe der Datei *suche_ausgabe.php* (die Tabelle mit den Ergebnissen) wird in den angegebenen Container (*#ausgabe*) eingefügt.

3.3.3.2 **Eingabefelder**

Die Tabellenzellen der Änderungstabelle enthalten jeweils ein <*span*> Element und ein Inputfeld. Wird auf den Text im <*span*> geklickt, wird das Inputfeld sichtbar und das <*span*> ausgeblendet. Wird das Inputfeld verlassen, wird es aus- und das <*span*> eingeblendet. Wenn der Nutzer Änderungen am Text vorgenommen hat, soll dies kenntlich gemacht werden.

Das Aus- und Einblenden wird über die Methode *.toggle()* gesteuert. Diese erkennt automatisch, ob das betreffende Element gerade sichtbar ist oder nicht und stellt die Sichtbarkeit über ein *„style“* Attribut entsprechend um.

Beim Erstellen des Feldes wird von der Methode *.data()* Gebrauch gemacht. Diese Methode ermöglicht es Daten zu einem Element zu speichern und gewissermaßen an das Element anzuhängen. Die gespeicherten Daten sind sodann überall im Script verfügbar, wenn auf das jeweilige Element zugegriffen wird. Hier wird dies genutzt, um den ursprünglichen Inhalt der Tabellenzelle zu speichern. Der Text wird unter dem Schlüssel *„originaltext“* abgelegt. Sobald ein <*span*> Teil eines Wrapper-Sets ist, kann mit *.data("originaltext")* auf den gespeicherten Text zugegriffen werden.

Wird das Feld verlassen (verliert den Fokus), wird geprüft, ob sich der aktuelle Inhalt vom Originaltext unterscheidet. Ist die nicht der Fall, wird der normale Tabellentext (im <*span*>) wieder eingeblendet. Hat sich der Text geändert, wird der geänderte Text angezeigt und zudem rot eingefärbt.

Eine einfache Prüfung auf eine Änderung über ein *onChange*-Event wäre hier nicht ausreichend. Es ist möglich, dass der Nutzer zuerst eine Änderung vornimmt, diese im weiteren Verlauf (nachdem das Feld bereits verlassen wurde) jedoch wieder zurücknimmt und den originalen Text in das Inputfeld einträgt. Da in diesem Fall effektiv keine Änderung erfolgte, soll der Text nicht rot dargestellt werden.

Jedes <*span*> und jedes Inputfeld bekommt einen Tabindex zugewiesen. Über einen Druck auf die Tab-Taste kann von einem Inputfeld zum nächsten <*span*> gesprungen werden. Dieses bekommt daraufhin den Focus und das dazugehörige Inputfeld wird eingeblendet. Der Tabindex muss beiden

Elementen zugeteilt werden, da von einem <*span*> zu einem Input gesprungen wird, auch wenn es für den Nutzer so aussieht, als fände der Wechsel zwischen zwei Inputfeldern statt. Zusammengehörenden <*span*> und Inputfeldern wird derselbe Tabindex zugewiesen um sicherzustellen, dass kein Feld übersprungen wird, da bei identischen Tabindizes die Reihenfolge im Code entscheidet und die beiden Elemente immer aufeinanderfolgen. Jedesmal wenn ein neues Feld hinzugefügt wird, werden die Tabindizes für die gesamte Tabelle neu vergeben. So fügen sich neue Elemente an der richtigen Position in der Reihenfolge ein.

3.3.3.3 **Hinzufügen und Entfernen von Feldern**

Werden neue Eingabefelder erzeugt, wird auf die versteckten Textfelder zurückgegriffen[1], um die Anzahl an bereits vorhandenen Feldern zu ermitteln und die Namen für die neuen Felder zu generieren. Die Werte werden nach dem Erstellen entsprechend erhöht. Werden neue Übersetzungen oder Phrasen hinzugefügt, wird für jedes Lemma ein eigenes Feld mit der dazugehörigen Anzahl an Übersetzungen / Phrasen erstellt.

Der HTML-Code für die neuen Tabellenzeilen und Inputfelder liegt bereits fertig in einer Variablen vor. Er wird lediglich noch um die zu vergebenden Namen und Nummern erweitert und in die Ausgabetabelle angefügt. Alle <*select*> Elemente im jeweiligen <*tbody*> werden um eine Option erweitert. Gab es vorher beispielsweise zwei Übersetzungen, enthielten die Dropdownmenüs die Werte *1* und 2. Wird nun eine neue Übersetzung hinzugefügt, sollen auch die bereits bestehenden Menüs die Werte *1*, *2* und *3* enthalten.

3.3.3.4 **Löschen**

Ist ein Inputfeld beim Verlassen leer, bedeutet dies, dass der Text aus der Datenbank gelöscht werden soll. Um dies dem Nutzer deutlich zu machen, wird der Text im <*span*> durchgestrichen (das Inputfeld wird leer übermittelt). Wird eine Übersetzung gelöscht, wird die dazugehörige Erläuterung ebenfalls entfernt (= das Geschwisterelement). Wird eine Phrase gelöscht, wird neben der Erläuterung auch die Übersetzung der Phrase gelöscht (= alle Inputfelder in derselben Zeile). Wird ein Lemma gelöscht, werden alle

[1] Siehe Abschnitt 3.2.2.2

dazugehörigen Begriffe gelöscht (= alle Inputfelder im <*tbody*> des Lemmas und alle im <*tbody*> der Phrase).

Der Text, der durchgestrichen angezeigt wird, ist immer der Originaltext. Wurde ein Begriff zuerst geändert und dann gelöscht, wird der ursprüngliche Text aus der Datenbank entfernt und nicht der neu eingegebene, dies soll dem Nutzer bewusst gemacht werden.

3.3.3.5 **Prüfung**

Bevor die Daten tatsächlich in die Datenbank geschrieben werden, sollen sie überprüft werden. Die Prüfung bezieht sich darauf, ob die Daten die Integrität der Datenbank oder die Struktur des Wörterbuches verletzen, nicht ob sie Sinn ergeben, richtig geschrieben oder richtig übersetzt sind. Die Dateneingabe ist nur eingeloggten Benutzern zugänglich, es wird davon ausgegangen, dass diese ein Interesse an einem funktionierenden System haben und nicht versuchen mutwillig Fehler zu verursachen. Aus diesem Grund wird es als unproblematisch angesehen, die Prüfung mittels JavaScript und damit auf dem Computer des Nutzers durchzuführen, wo sie theoretisch umgangen oder manipuliert werden könnte.

Die Prüfung soll den Nutzer bei der Arbeit unterstützen und ihn nicht bevormunden oder versuchen für ihn zu denken. Es wird nur geprüft, ob benötigte Felder ausgefüllt sind oder ob Daten doppelt vorkommen. Felder in denen beispielsweise nur Leerzeichen stehen gelten im Sinne der Prüfung als ausgefüllt. Solche absichtlichen Falscheingaben sollen aber auch nicht gefunden werden.

Bevor die Funktion *speichern* den Inhalt der Inputfelder an *speichern.php* schickt, werden die beiden Prüfungsfunktionen *check* und *doppelt* ausgeführt. Die Prüfung gliedert sich in zwei Schritte: Zuerst wird überprüft, ob alle benötigten Felder ausgefüllt sind (check) danach, ob doppelte Eingaben vorliegen (doppelt).

Die Funktion *check*: Sind bestimmte Felder ausgefüllt, so müssen auch gewisse andere Felder Daten enthalten: Eine Erläuterung erfordert immer ein Feld welches erläutert wird. Eine Übersetzung erfordert ein Lemma. Eine Phrase erfordert ein Lemma und eine Übersetzung der Phrase. Ist ein Feld leer, so soll es gelöscht werden. In diesem Fall übernimmt die Funktion zum Löschen automatisch die Prüfung, da alle voneinander abhängigen Felder

zusammen gelöscht werden. Für die weitere Prüfung wird nun noch zwischen einem Feld mit Platzhaltertext (*nicht ausgefüllt*) und einem *ausgefüllten* Feld unterschieden. Existiert ein nicht ausgefülltes Lemma-Feld, dürfen in seinem <*tbody*> keine weiteren Felder ausgefüllt sein. Ist ein Phrasen-Feld ausgefüllt, so darf das Feld für die Übersetzung der Phrase keinen Platzhalter enthalten. Ist ein Erläuterungs-Feld ausgefüllt, darf das übergeordnete Feld keinen Platzhalter enthalten.

Ist eine der Bedingungen nicht erfüllt wird die Tabellenzelle in der eine Eingabe fehlt gelb gefärbt und die Variable *error* auf den wert *true* gesetzt. Der Inhalt dieser Fehlervariable wird an die Funktion *speichern* übermittelt, die im Falle eines Fehlers abbricht und nicht speichert.

Die Funktion *doppelt*: Lemmata dürfen mit einer Bedeutung nur einmal in der Datenbank stehen, eine Kombination aus Lemma und Erläuterung muss demnach einmalig sein. Diese Funktion gliedert sich in zwei Teile, zum einen wird geprüft, ob bei der Eingabe doppelte Begriffe verwendet wurden, zum anderen wird geprüft, ob neu eingegebene Begriffe bereits in der Datenbank gespeichert sind. Um auf doppelt eingegebene Begriffe zu prüfen, wird für jedes Lemma ein zusammenhängender String aus dem Lemma und der Erläuterung gebildet und in einem Array abgelegt. Immer wenn ein neuer String hinzugefügt wird, wird geprüft, ob er schon im Array vorhanden ist. Ist dies der Fall, liegt eine doppelte Eingabe vor und ein Fehler wird ausgegeben.
Um zu prüfen, ob neu erfolgte Einträge bereits in der Datenbank gespeichert sind, erfolgt nacheinander eine Datenbankabfrage für jeden Begriff (über die Datei *doppelt.php*). Das Ergebnis dieser Abfragen wird in einen Container geschrieben. Die Funktion *speichern* liest den Containerinhalt aus. Ist er leer, gibt es keine doppelten Begriffe und der Speichervorgang wird fortgesetzt. Ist er nicht leer, wird der Containerinhalt (die doppelten Begriffe) ausgegeben.
Die Kommunikation mit dem Server für die Datenbankabfrage findet hier synchron statt, im Gegensatz zu regulärem (asynchronen) Ajax. Auf diese Weise ist sichergestellt, dass die Antwort des Servers auch wirklich vorliegt, bevor auf ihr Vorhandensein hin geprüft wird. Die daraus resultierende Verzögerung im Ablauf (das Script wird erst weiter ausgeführt, wenn alle Abfragen ausgeführt sind) ist vertretbar. Der Nutzer hat an dieser Stelle einen

Arbeitsschritt abgeschlossen und auf *speichern* geklickt, eine kurze Wartezeit ist hier nicht ungewöhnlich und unterbricht keine Arbeitsvorgänge.

War der Speichervorgang erfolgreich, erhält der Nutzer eine Erfolgsmeldung. Wurde versucht zu speichern, ohne dass Änderungen vorgenommen worden sind (kein Element mit der Klasse *„neu“* vorhanden), wird dies angezeigt. Dies soll verhindern, dass ein Nutzer glaubt Daten würden geändert/gelöscht, obwohl dies nicht der Fall ist, z. B. aufgrund eines Bedienfehlers.

3.3.3.6 **Autocomplete**

Die Suchvorschläge werden durch die jQuery Methode *autocomplete* erzeugt. Diese ist Teil des *jQuery UI Plug-ins* und wird auf das Suchfeld angewendet. Immer wenn sich der Text im Suchfeld ändert, wird diese Methode aufgerufen und erstellt eine neue Liste mit Vorschlägen.

Sie gliedert sich in die drei Teile *source*, *success* und *select*. Unter *source* wird die Verbindung zu *autocomplete.php* aufgebaut und die zu übermittelnden Daten festgelegt. Diese werden hier im JSON-Format[1] übertragen und empfangen, dies ist eine Voraussetzung von *autocomplete*. *Success* bestimmt, was bei einer erfolgreichen Ajax-Abfrage, was zu tun ist. Hier soll das Abfrageergebnis Element für Element durchgegangen werden und alle Begriffe ausgegeben werden. Es muss nur angegeben werden, welche Teile der Serverantwort verwendet werden sollen. Die Liste an sich wird von der *autocomplete* Funktion automatisch erstellt. *Select* legt das Verhalten fest, wenn ein Eintrag aus der Liste gewählt wird. Hier soll er in das Suchfeld übernommen und eine Suche danach ausgeführt werden.

3.3.3.7 **Tabs**

Für die Tabs wird auf die *jQuery UI*-Methode *.tabs()* zurückgegriffen. Diese stellt alle Links in einem definierten Container per CSS zu vertikalen Tabs um. Ein Klick auf einen der Links ruft die im Link genannte Datei *(liste.php*) auf und stellt das Ergebnis unterhalb der Tabs dar. Die genaue Struktur wird automatisch von *.tabs()* erzeugt. Es müssen nur die zu verarbeitenden Links angegeben werden.

[1] Siehe Abschnitt 3.2.1.5

liste.php bekommt über den Link einen Buchstaben übermittelt und sucht alle hochdeutschen Lemmas, die mit diesem Buchstaben beginnen[1]. Die Ergebnisse der Abfrage werden wiederum jeweils in einen Link eingebettet und ausgegeben. Ein Klick auf einen solchen Listeneintrag führt eine Suche nach dem jeweiligen Begriff aus.

3.3.3.8 **Suchwort hervorheben**

Um den Suchbegriff fett hervorzuheben kommt das Plug-In *„Highlight“* zum Einsatz. Dieses durchsucht das Zielelement (hier die Ausgabetabelle) nach einem String (hier der Inhalt des Suchfeldes) und umschließt ihn mit einem der Klasse *„highlight“*. Über die entsprechende CSS Datei wird eingestellt, dass diese Klasse fett darzustellen ist.

Sollen die Phrasen und Erläuterungen nicht mit durchsucht werden, so soll in ihnen auch das Suchwort nicht hervorgehoben werden. In diesem Fall wird, wie oben, zuerst ein Highlight gesetzt, dieses aber sofort wieder durch die Methode *.removeHighlight()* entfernt (vgl. jQuery Highlight 2007).

3.3.3.9 **Vorschau und Dialoge**

Die pop-up Fenster für die Vorschauansicht[2] und die Fehlermeldungen bei der Datenprüfung werden durch das UI-Plug-in erzeugt. Hierzu wird der gewünschte Text (das Ergebnis der Ajax abfragen) in ein <*div*> eingefügt und auf dieses die Methode *.dialog()* mit dem Parameter *„open“* angewandt. Das Plug-in erzeugt nun aus dem im <*div*> enthaltenen Text eine Dialogbox.

Damit dies funktioniert, muss dem UI-Plug-in mitgeteilt werden, dass es sich bei dem Zielelement um eine Quelle für eine Dialogbox handelt. Dies passiert durch das Aufrufen der Methode *.dialog()* beim Laden des Dokumentes. Dieser werden, die den Dialog betreffenden, Optionen als Parameter übergeben. Hier ist es wichtig, die Option *„autoOpen: false“* zu setzen, andernfalls öffnet sich der Dialog beim Laden automatisch und kann später nicht mehr zuverlässig durch einen Event geöffnet werden.

[1] Siehe Abschnitt 3.2.1.6
[2] Siehe Abschnitt 3.2.2.6

Unter der Option *„buttons“* werden die Buttons im Format *„Beschriftung: Funktion beim Klick*“ definiert.

```
buttons: {
   "OK":function(){
      $(this).dialog("close");}
   }
```

3.4 CSS und Design

Das Design war durch die Gestaltung der alten *inslex*-Version vorgegeben. Um es umzusetzen, wurden die Werte aus der bestehenden CSS-Datei übernommen und die Klassenbezeichnungen in den an die tatsächlich verwendeten angepasst. Für die Suche und die Ausgabetabelle gibt es zwei verschiedene Stylesheets. Auch wenn sie sich teilweise überschneiden sind die Unterschiede doch groß genug, um diesen Weg zu wählen, und somit Konflikte zu vermeiden. Elemente, die auf beiden Seiten gleich aussehen haben dieselben Bezeichnungen, sowohl im HTML, als auch in den beiden CSS-Dateien. Anpassungen, die beide Seiten betreffen sollen, sind so leicht vorzunehmen, gleichzeitig kann jede Seite individuell gestaltet werden, ohne in die Gefahr zu laufen die andere zu beeinflussen.

Die vom jQuery UI-Plug-in erzeugten Elemente werden durch eigene Stylesheets gestaltet. Um sie anzupassen stellt das Plug-in auf seiner Webseite ein Online-Tool (*Theme-Roller*) (jQuery UI Themeroller 2011) zur Verfügung. Hier können die einzelnen Elemente über eine graphische Benutzeroberfläche angepasst werden. Die CSS-Dateien werden automatisch erzeugt und zum Download bereitgestellt. Manuelle Änderungen an den automatisch erzeugten Dateien wurden im Bereich der Tabs vorgenommen, um die Größe der einzelnen Tabs anzupassen. Die Reihe der Tabs darf nicht die gesamte Seitenbreite beanspruchen, je nach verwendeten Browser, kann es sonst zu einem Umbruch innerhalb der Tabs kommen. Auf ein Zusammenführen mit dem restlichen Stylesheet wurde verzichtet, um eine klare Trennung der manuell und automatisch generierten Dateien zu gewährleisten.

3.5 Sicherheit und Benutzerverwaltung

Alle in der Datenbank gespeicherten Daten sollen für alle Besucher der Webseite einsehbar sein. Das Sicherheitskonzept beschränkt sich demnach

darauf, dass nur befugte Personen in der Lage sein sollen, die Daten ändern zu können.

Im Front-End wird dies erreicht, indem Steuerzeichen aus der Benutzereingabe gefiltert werden, um MySQL-Injection Angriffe zu verhindern. Für die über Radio-Buttons und Checkboxen übermittelten Optionen sind erlaubte Werte festgelegt. Stimmt der Inhalt der Optionsvariablen nicht mit diesen Werten überein, wird er auf einen Standardwert gesetzt. So soll z. B. verhindert werden, dass über eine manuelle Manipulation der *Get*-Variablen in der URL Code eingeschleust wird.

Der Zugang zum Back-End wird über das Content-Management-System der INS Webseite geregelt. Da hier nur sehr wenige Personen Zugang haben sollen, wird ein Zugang für alle Beteiligten eingerichtet, so dass immer nur ein Benutzer gleichzeitig in das Back-End eingeloggt sein kann. Auf diese Weise wird verhindert, dass es zu Problemen durch das gleichzeitige Ändern von Daten kommt. Vom Back-End aus ist der Zugang auf die komplette Datenbank möglich. Da das Löschen und Verändern von Einträgen bereits über die regulären Funktionen möglich ist, liegt der Schwerpunkt der Sicherheit in diesem Bereich darauf, zu verhindern, dass Einträge so manipuliert werden, dass ihre Anzeige über das Front-End negative Effekte hat. So werden beispielsweise HTML-Zeichen nur maskiert in die Datenbank geschrieben, damit der Code beim Anzeigen des betreffenden Artikels nicht ausgeführt wird.
Weitere Einzelheiten hierzu finden sich als Kommentar im Code.

3.6 Datenbankanbindung

Die Login- und Verbindungsdaten zur Datenbank sind in der Datei *inslex_config.php* gespeichert. Diese Datei wird über einen *include*-Befehl in die PHP-Dateien eingefügt. Änderungen an den Zugangsdaten können so zentral an einer Datei vorgenommen werden. Zusätzlich wird in dieser Datei die Datenbankverbindung auf den Zeichensatz UTF-8 gestellt.

Da die Datei *inslex_config.php* selbst keine Ausgabe produziert, werden bei einem direkten Aufruf der Datei auch keine Daten an den Client übermittelt (der Server greift auf die Datei zu, der Client erhält nur die ausgegebenen Daten). Somit sind die Logindaten der Datenbank geschützt.

Es ist denkbar diese Datei in Zukunft noch weiter auszubauen. Empfehlenswert ist es z. B. die Texte für Fehlermeldungen hier zentral abzulegen. Für Laien ist es deutlich einfacher eine solche (übersichtliche) Konfigurationsdatei zu bearbeiten, anstatt die Änderungen direkt im Quellcode vornehmen zu müssen. Auch können hier die Pfade zu den einzelnen Dateien als Variablen gespeichert und in den Scripten selbst mit diesen Variablen gearbeitet werden. Ändert sich ein Dateiname, hat dies den Vorteil, dass die Änderung nur einmal vorgenommen werden muss. In diesem Fall ist darauf zu achten, die an die PHP Dateien übermittelten Variablen in JavaScript verfügbar zu machen (für Ajax-Abfragen), etwa indem sie in versteckte Textfelder geschrieben und diese durch JavaScript ausgelesen werden.

3.7 Einbindung in die bestehende Seite

Die Frage der Einbindung in das Typo3-System der bestehenden Webseite des INS ist bei Abschluss dieser Arbeit noch nicht endgültig geklärt, hier bedarf es weiterer Informationen seitens des aktuellen Serverbetreibers.

Eine Möglichkeit ist, dass die zu *inslex* gehörenden Webseiten unabhängig von der bestehenden Seite gespeichert werden und über ein iFrame angezeigt werden. Die bisherige Version ist auf diese Weise eingebunden. Das iFrame hat eine Breite von 749 und eine Höhe von 550 Pixeln. Die Gestaltung von *inslex* wurde auf diese Abmessungen hin optimiert.

Eine andere Möglichkeit besteht darin die *inslex*-Seiten als Typo3 Unterseiten, bzw. als eine Extension einzubinden. Diese Lösung bedeutet einen Mehraufwand, hat aber den Vorteil, dass *inslex* so vollkommen in das Content-Management-System integriert ist.

3.8 Barrierefreiheit

Die Suche ist komplett über die Tastatur zu bedienen. Das Anwählen von Eingabemöglichkeiten funktioniert über die Tab-Taste, Sucheingaben können mit Enter abgeschickt werden und das Auswählen von Suchvorschlägen ist mit den Pfeiltasten möglich.

JavaScript ist inzwischen so weit verbreitet, dass die Seite von aktiviertem JavaScript ausgeht. Solle es dennoch deaktiviert sein, ist es dennoch möglich die Seite zu benutzen. In diesem Fall aber wesentlich unkomfortabler, da die Ajax-Abfragen entfallen und die Suchergebnisse auf einer neuen

Seite präsentiert werden und durch JavaScript vergebene Gestaltungelemente fehlen. Die Artikelstruktur bleibt jedoch erhalten, eine Zuordnung von Lemma zu den Übersetzungen ist weiterhin möglich. Über diesen (Um)Weg ist damit gleichzeitig eine Möglichkeit gegeben, die Ausgabe als reine Textversion zu gestalten. Die erweiterten Suchoptionen werden beim Laden angezeigt und erst durch JavaScript ausgeblendet. Ist JavaScript deaktiviert stehen die erweiterten Optionen somit trotzdem zur Verfügung.

Fazit und Ausblick

Die Arbeit an *inslex* hat gezeigt, dass es möglich ist, ein Onlinewörterbuch mittels der verwendeten Techniken umzusetzen. Die Ähnlichkeit der Mikrostruktur eines Wörterbuches mit der Struktur einer relationalen Datenbank war von großer Hilfe. Es ist deutlich geworden, dass es empfehlenswert ist, diese Strukturen genau zu planen. Das Finden von Gemeinsamkeiten zwischen dem Wörterbuch und einer Datenbank war ein großer Teil der Aufgabe.

Im Laufe der Arbeit hat sich gezeigt, dass die Entwicklung der eigentlichen Datenbank nicht alles ist. Die beste Datenbank nützt nichts, wenn die dazugehörige Benutzeroberfläche ihrer nicht gerecht wird. Die Herausforderung bestand auch darin, die grundlegenden Wörterbuchstrukturen über die Oberfläche abzubilden und dem Nutzer auch so zu vermitteln. Die Bedienung von *inslex* soll möglichst so einfach sein, dass es intuitiv genutzt werden kann und die Konzentration auf der eigentlichen Aufgabe liegen kann und nicht auf der Bedienung der Werkzeuges. Ich bin zuversichtlich, dass *inslex* einen Beitrag zur Lebendigkeit und Aktualität des Plattdeutschen liefern kann.

Die neue Version von *inslex* greift die Daten der bisherigen Version auf und präsentiert sie neu. Die geplanten Ansprüche wurden erfüllt. Es ist nun leichter gezielt nach einem Begriff zu suchen, die Zahl der nicht-relevanten Treffer konnte durch die Beschränkung der durchsuchten Felder reduziert werden. Trotzdem ist es weiterhin möglich alle gespeicherten Daten zu finden, wenn dies gewünscht ist. Die Beziehungen zwischen den einzelnen Artikelteilen konnten durch die Aufteilung der Mikrostruktur auf Übersetzungen, Erläuterungen und Phrasen verbessert werden. Es ist klar ersichtlich,

welche Übersetzungen zu welcher Bedeutung gehören. Während *inslex* in der bisherigen Version, wie der Name sagte, eine *Wortliste* war, hat es sich nun Richtung *Wörterbuch* entwickelt.

Das Projekt ist in sich abgeschlossen. Trotzdem bietet *inslex* noch zahlreiche Möglichkeiten für Erweiterungen und Verbesserungen, von vielen Ideen ließen sich leider nicht alle gleich umsetzten. Denkbar ist es etwa das Änderungsdatum der Einträge mit zu speichern. Dies würde es z.B. ermöglichen sich alle neuen Artikel des letzten Monats anzeigen zu lassen. Die Suche könnte so gestaltet werden, dass Verweise zwischen den Artikeln möglich sind und ein Klick auf einen Begriff eine Suche nach diesem startet. Das Löschen von Einträgen würde an Nutzerfreundlichkeit gewinnen, wenn gelöschte Einträge nur markiert und nicht entfernt werden und das Löschen somit rückgängig zu machen ist. Auch Erweiterungen bei den gespeicherten Daten Sind denkbar, etwa das Einbeziehen von zusätzlichen Informationen. *Inslex* bietet die Möglichkeit, solche Erweiterungen in das System zu integrieren.

Literaturverzeichnis

Atkins 2008
ATKINS, Sue B. T. ; Rundell, Michael: *The Oxford Guide to Practical Lexicography.* Oxford : Oxford University Press, 2008. – ISBN 978-0-19-927771-1

Bargstedt 2009
BARGSTEDT, Stefan: *Platt! : Wo und wie Plattdeutsch ist*. 2., überarb. Aufl. Bremen : Schünemann, 2009. – ISBN 978-3-7961-1907-1

Bergenholtz / Nielsen /Tarp 2009
BERGENHOLTZ, Henning (Hrsg.) ; NIELSEN, Sandro (Hrsg.) ; TARP, Sven (Hrsg.): *Lexicography at a Crossroads : Dictionaries and Encyclopedias Tody, Lexicographical Tools Tomorrow*. Bern : Lang, 2009 (Linguistic Insights 90). – ISBN 978-3-03911-799-4

Bergenholtz / Tarp 1995
BERGENHOLTZ, Henning (Hrsg.) ; TARP, Sven (Hrsg.): *Manual of Specialised Lexicography*. Amsterdam : Benjamins, 1995. – ISBN 90-272-1612-6

Carstensen et al. 2004
CARSTENSEN, Kai-Uwe (Hrsg.) ; EBERT, Christian (Hrsg.) ; ENDRISS, Cornelia (Hrsg.) ; JEKAT, Susanne (Hrsg.) ; KLABUNDE, Ralf (Hrsg.) ; LANGER, Hagen (Hrsg.): *Computerlinguistik und Sprachtechnologie : Eine Einführung*. 2., überab. und erw. Aufl. München : Spektrum, 2004. – ISBN 3-8274-1407-5

Conrad et al. 1985
CONRAD, Rudi (Hrsg.) ; BARTSCHAT, Brigitte ; CONRAD, Rudi ; HEINEMANN, Wolfgang ; PFEIFER, Gerlinde ; STEUBER, Anita: *Lexikon Sprachwissenschaftlicher Termini*. Leipzig : Bibliographisches Institut, 1985

Fehrs-Gilde 2004
FEHRS-GILDE VEREIN ZUR FÖRDERUNG DES NIEDERDEUTSCHEN E.V.: *der neue SASS: Plattdeutsches Wörterbuch*. 3., überarb. Auflage. Neumünster : Wachtelholz, 2004. – ISBN 3-529-03000-7

Geeb / Spree 2004
GEEB, Franziskus ; SPREE, Ulrike: *Wörterbücher und Enzyklopädien*. In: KUHLEN, Rainer (Hrsg.) ; SEEGER, Thomas (Hrsg.) ; STRAUCH, Dietmar (Hrsg.): *Grunglagen der praktischen Information und Dokumentation*. Bd. 1 : *Handbuch zur Einführung in die Informationswissenschaft und -praxis*. 5., völlig neu gefasst Ausgabe. München : Saur, 2004. – ISBN 3-598-11674-8. S. 481 – 492

Hearst 2009
HEARST, Marti A.: *Search User Interface*. Cambridge : Cambridge University Press, 2009. – ISBN 978-0-521-11379-3

INS Homepage 2011
INSTITUT FÜR NIEDERDEUTSCHE SPRACHE (Hrsg.): *Startseite*. URL http://www.ins-bremen.del. – Abrufdatum: 30.03.2011

INS inslex 2011
INSTITUT FÜR NIEDERDEUTSCHE SPRACHE (Hrsg.): *INS-lex : Wortliste plattdeutsche Nachrichten*. URL http://www.ins-bremen.de/de/service/ins-lex/wortliste-suche.html. – Abrufdatum: 30.03.2011

INS Satzung 2005
INSTITUT FÜR NIEDERDEUTSCHE SPRACHE (Hrsg.): *Satzung*. URL http://www.ins-bremen.de/de/das-ins/aufgaben/satzung-und-vorstand-mitgliedschaft.html. – Aktualisierungsdatum: 24.11.2005. – Abrufdatum: 30.03.2011

jQuery Dokumentation 2011
JQUERY PROJECT (Hrsg.):*Documentation : Main Page*. URL: http://docs.jQuery.com/Main_Page. – Abrufdatum: 30.03.2011

jQuery Highlight 2007
BURKARD, JOHANN: *highlight : JavaScript text higlighting jQuery plugin*. URL: http://johannburkard.de/blog/programming/javascript/highlight-javascript-text-higlighting-jquery-plugin.html. – Aktualisierungsdatum: 15.09.2007. – Abrufdatum: 30.03.2011

jQuery UI 2011
JQUERY PROJECT (Hrsg.) ; JQUERY UI TEAM (Hrsg.): *jQuery user interface*. URL: http://jqueryui.com. – Abrufdatum: 30.03.2011

jQuery UI Themeroller 2011
JQUERY PROJECT (Hrsg.) ; JQUERY UI TEAM (Hrsg.): *jQuery user interface : ThemeRoller*. URL: http://jqueryui.com/themeroller/. – Abrufdatum: 30.03.2011

Kannengiesser 2005
KANNENGIESSER, Matthias: *Webseiten mit PHP 5 & MySQL 4. : Lösungen für Ein- und Umsteiger*. München : Markt + Technik, 2005. – ISBN 3-8272-6867-2

Knoop 1997
KNOOP, Ulrich: *Wörterbuch der deutschen Dialekte : Eine Sammlung von Mundartwörterbüchern aus zehn Dialektgebieten im Einzelvergleich, in Sprichwörtern und Redewendungen*. Gütersloh : Bertelsmann Lexikon, 1997. – ISBN 3-577-10574-7

Knorz 1997
KNORZ, Gerhard: *Datenbank: Entwurfsmethoden*. In: BUDER, Marianne (Hrsg.) ; REHFELD, Werner (Hrsg.); SEEGER, Thomas (Hrsg.) ; STRAUCH, Dietmar (Hrsg.): *Grundlagen der praktischen Information und Dokumentation : Ein Handbuch zur Einführung in die fachliche Informationsarbeit*. Bd. 2. 4. völlig neu gefasste Ausgabe. München : Saur, 1997. – ISBN 3-598-11309-9. S. 664 – 678

Lemberg 2001
LEMBERG, Ingrid: *Aspekte der Online-Lexikographie für wissenschftliche Wörterbücher*. In: LEMBERG, Ingrid (Hrsg.) ; BERNHARD, Schröder (Hrsg.) ; STORRER, Angelika (Hrsg.): *Chancen und Perspektiven computergestützter Lexikographie : Hypertext, Internet und SGML/XML für die Produktion und Publikation digitaler Wörterbücher*. Tübingen : Niemeyer, 2001 (Series Maior 107) . – ISBN 3-484-39107-3. S. 71-92

Lindley 2010
LINDLEY, Cody: *Grundlagen von jQuery*. In: JQUERYCOMMUNITY EXPERTS (Hrsg.) ; DEMMING, Thomas (Übers.): *jQuery Kochbuch*. Bejing : O'Reilly, 2010. – ISBN 978-3-89721-999-1

Lindow et al. 1998
LINDOW, Wolfgang ; MÖHN, Dieter ; HERMANN Niebaum ; STELLMACHER, Dieter ; TAUBKEN, Hans ; WIRRER, Jan: *Niederdeutsche Grammatik*. Bremen : Schuster Leer, 1998 (Schriften des Instituts für niederdeutsche Sprache) (Reihe: Dokumentation Nr. 20). – ISBN 3-7963-0332-3

MySQL InnoDB 2011
ORACLE CORPORATION (Hrsg.): *MySQL 5.1 Reference Manual : 13.6. The InnoDB Storage Engine*. URL: http://dev.mysql.com/doc/refman/5.1/en/innodb-storage-engine.html. – Abrufdatum: 30.03.2011

MySQL MyISAM 2011
ORACLE CORPORATION (Hrsg.): *MySQL 5.1 Reference Manual : 13.5. The MyISAM Storage Engine*. URL: http://dev.mysql.com/doc/refman/5.1/en/myisam-storage-engine.html. – Abrufdatum: 30.03.2011

Nielsen / Loranger 2006
NIELSEN, Jakob ; LORANGER, Hoa: *Prioritizing Web Usability*. Berkeley : New Riders, 2006. – ISBN 0-321-35031-6

Reese / Yarger / King 2003
REESE, George ; YARGER, Randy ; KING, Tim: *MySQL : Einsatz und Programmierung*. 2. Aufl. Beijing : O'Reilly, 2003. – ISBN 3-89721-178-5. S. 1-34

Schaeder / Bergenholtz 1994
SCHAEDER, Burkhard (Hrsg.) ; BERGENHOLTZ Henning (Hrsg.): *Fachlexikographie : Fachwissen und seine Repräsentation in Wörterbüchern*. Tübingen : Narr, 1994. – ISBN 3-8233-4534-6

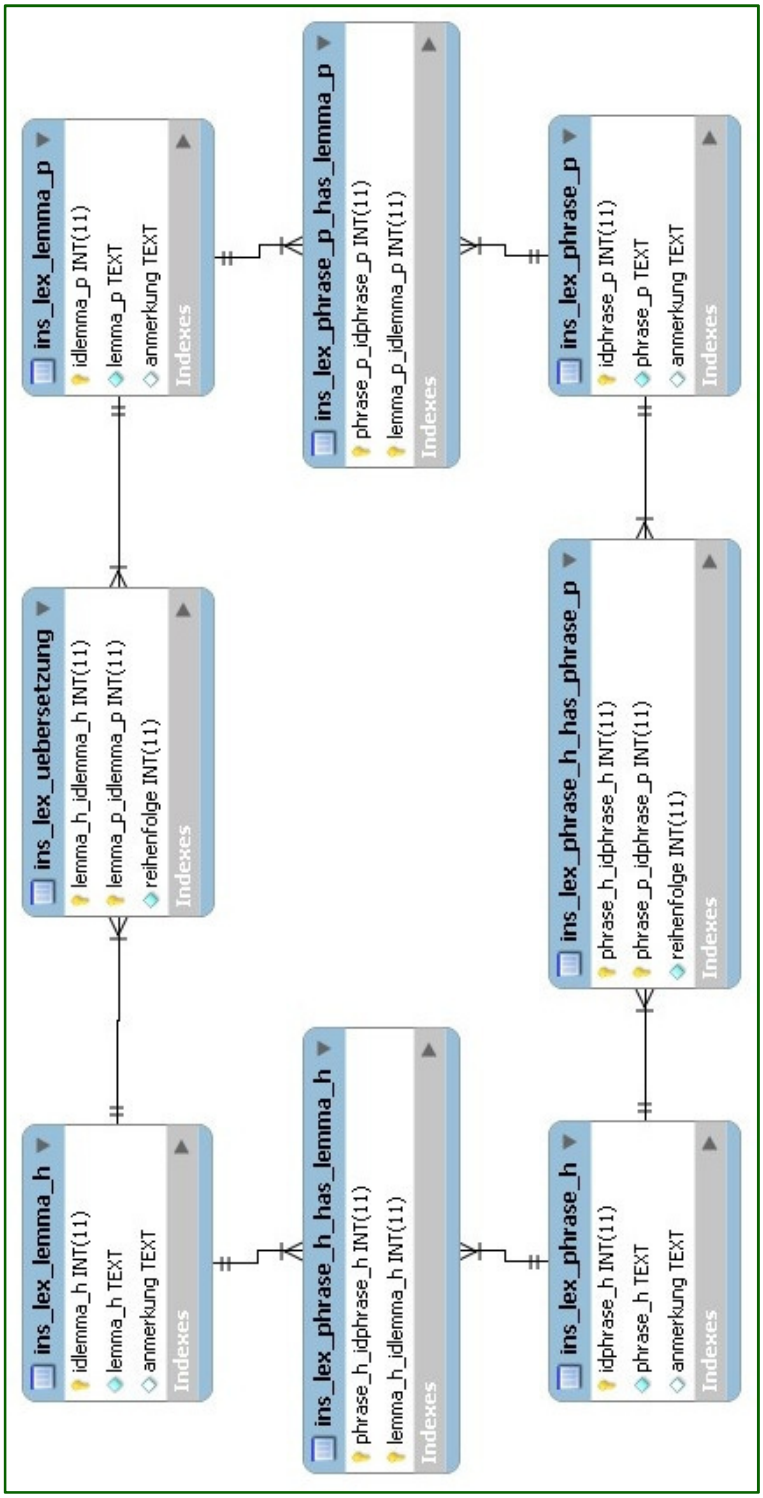
Schwarze / Wunderlich 1985
SCHWARZE, Christoph (Hrsg.) ; WUNDERLICH, Dieter (Hrsg.): *Handbuch der Lexikologie*. Königstein/Ts. : Athenäum, 1985. – ISBN 3-7610-8331-9

Schlaefer 2002
SCHLAEFER, Michael: *Lexikologie und Lexikographie : Eine Einführung am Beispiel deutscher Wörterbücher*. Berlin : Schmidt, 2002 (Grundlagen der Germanistik 40). – ISBN 3-503-061-43-6

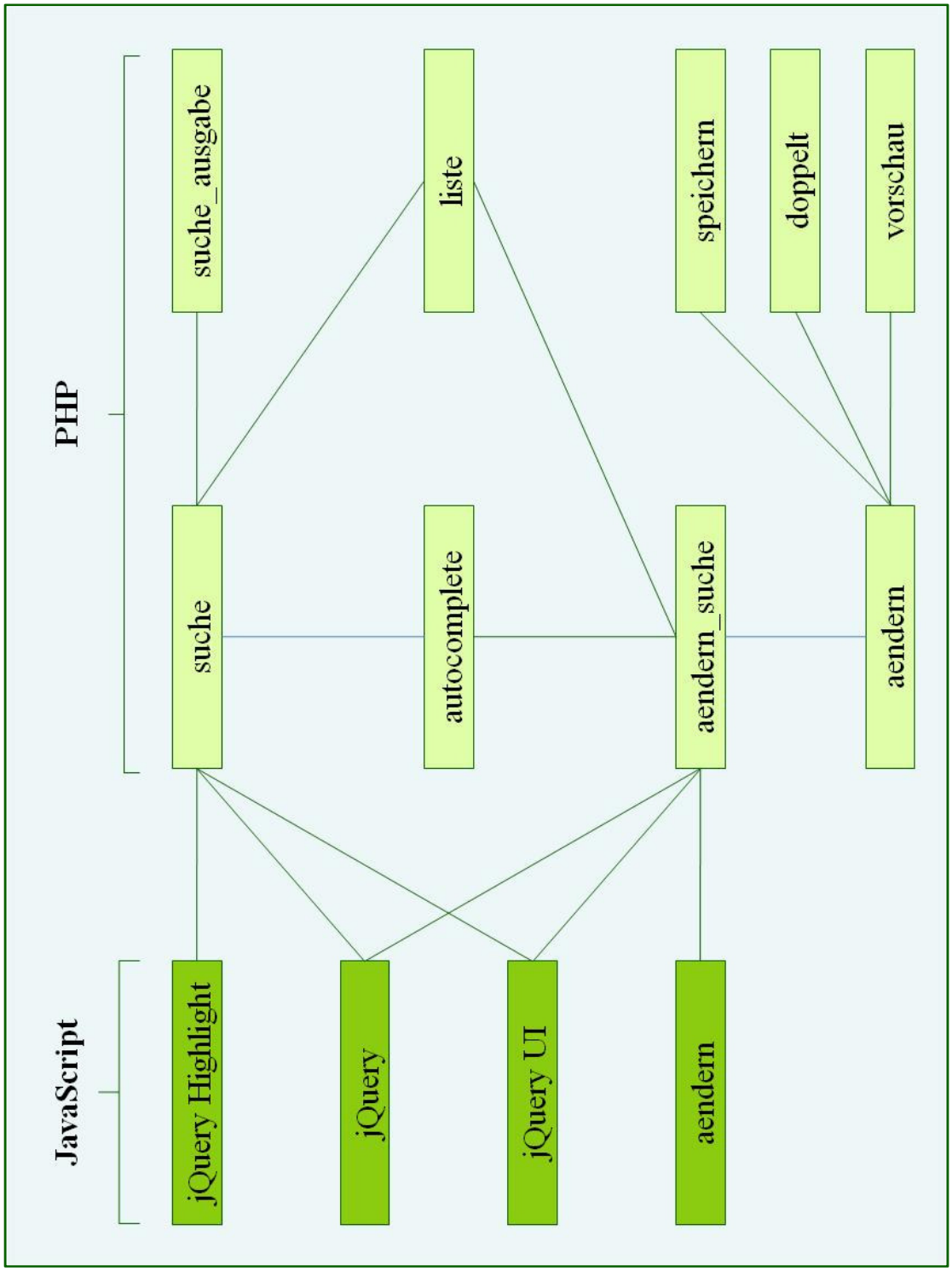
Zurawski 2007
ZURAWSKI, Nils: *Plattdeutsch digital: Formen der Sprach- und Identitätskonstruktion im Internet*. In: INSTITUT FÜR NIEDERDEUTSCHE SPRACHE (Hrsg.) ; VEREINIGUNG QUICKBORN (Hrsg.): *Kulturraum und Sprachbilder : Plattdeutsch gestern und morgen*. Bremen : Schuster Leer, 2007 (Schriften des Instituts für Niederdeutsche Sprache) (Reihe: Dokumentation Nr. 32). – Beiträge zum Symposium des Instituts für niederdeutsche Sprache und der Vereinigung Quickborn am 23. Oktober 2004 in Hamburg. – ISBN 978-3-7963-0377-7. S. 147 – 166

Anhänge

Er-Diagramm



Dateibeziehungen



CD mit Quellcode

(Der Quellcode ist nicht für eine Veröffentlichung bestimmt. Die CD ist nicht in der Bibliotheks-Version der Arbeit enthalten. Der Quellcode liegt den Prüfern und dem INS vor.)

CD-Inhalt

README.txt
db_inslex.sql

inslex
- aendern.php
- aendern_suche.php
- autocomplete.php
- doppelt.php
- frame.php
- frame_aendern.php
- liste.php
- speichern.php
- suche.php
- suche_ausgabe.php
- vorschau.php

css
- *images*
- inslex_aendern.css
- inslex_suche.css
- jquery-ui-1.8.9.inslex.css

js
- aendern.js
- jquery.highlight-3.js
- jquery-1.4.4.min.js
- jquery-ui-1.8.9.inslex.min.js

login
- inslex_config.php

Eidesstattliche Versicherung

Ich versichere, die vorliegende Arbeit selbstständig ohne fremde Hilfe verfasst und keine anderen Quellen und Hilfsmittel als die angegebenen benutzt zu haben. Die aus anderen Werken wörtlich entnommenen Stellen oder dem Sinn nach entlehnten Passagen sind durch Quellenangabe kenntlich gemacht.

Felix Vollmuth