

Hochschule Osnabrück

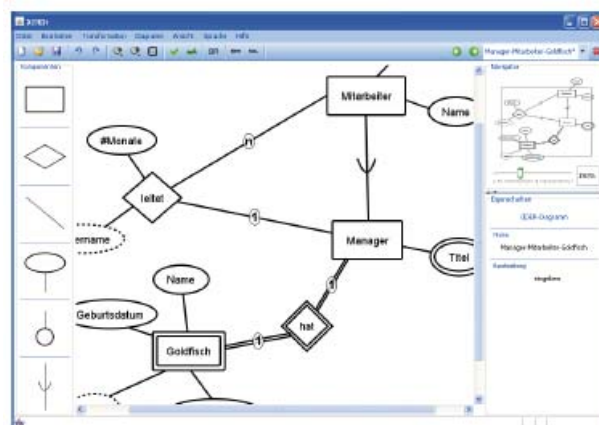
University of Applied Sciences

Professur für Datenbanken und Software-Entwicklung

XERDi

eXtended Entity-Relationship-Designer

Version 2.0.0 - geeignet für Java 7



Prof. Dr.-Ing. Heiko Tapken

Osnabrück, 17.09.2011

Anregungen bitte an:

xerdi@tapken-online.de

Inhaltsverzeichnis

1	Einleitung	1
2	Inbetriebnahme	2
3	Programmfunktionen	3
4	Bedienung	4
4.1	Quickstart	4
4.2	Diagrammerstellung	5
4.2.1	Anlegen eines neuen Diagramms	5
4.2.2	Öffnen eines Diagramms	5
4.2.3	Speichern von Diagrammen	5
4.2.4	Schließen von Diagrammen	6
4.2.5	Hinzufügen von Komponenten	6
4.2.6	Selektieren von Komponenten	7
4.2.7	Verschieben von Komponenten	7
4.2.8	Verbinden von Komponenten	7
4.2.9	Löschen von Komponenten	8
4.2.10	Editieren der Eigenschaften	8
4.2.11	Domäne eines Attributs anpassen	8
4.3	Transformation	9
4.3.1	EER zu ER	9
4.3.2	SQL-Generierung	9
4.4	Funktionen	10
4.4.1	Diagrammvalidierung	11
4.4.2	Kopieren und Einfügen	11
4.4.3	Verwaltung mehrerer Diagramme	11
4.4.4	Änderungshistorie	11
4.4.5	Autosave	12
4.4.6	Export	12
4.4.7	Zoom	12
4.4.8	Ausblenden von Attributen	12
5	Kompatibilität	13
A	Anhang	14
A.1	Tastenbelegungen und -kombinationen	14

A.2 Datenbankentwurf und Datenmodellierung	16
A.2.1 Phasen des Datenbankentwurfs	16
A.2.2 Das Entity-Relationship-Modell	18
A.2.3 Das relationale Modell	22
A.2.4 Die Datenmodellabbildung	23

1 Einleitung

XERDi ist ein Werkzeug zur grafischen Erstellung von (erweiterten) Entity-Relationship-Diagrammen unter Verwendung der Notation von ELMASRI und NAVATHE [EN04] sowie zur Generierung von SQL-Skripten der geläufigsten SQL-Dialekte aus den erstellten Diagrammen.

Es ist das Produkt der zwei Individuellen Projekte „Entwicklung einer Komponente zur grafischen Erstellung von erweiterten Entity-Relationship-Diagrammen“ von Kiril Schröder [Sch06] sowie „Automatisierung der Datenmodellabbildung zwischen (E)ER-Modell und relationalem Modell“ von Christian Burmeister [Bur06]. Ausgeschrieben wurden beide Projekte von der Abteilung Informationssysteme des Departements für Informatik der CvO Universität Oldenburg mit dem vorrangigen Ziel, den Lehrbetrieb der Veranstaltung Informationssysteme 1 zu unterstützen. Die Bedienung und der Funktionsumfang sind daher auf die diese Veranstaltung besuchenden Studenten zugeschnitten.

Anhand der Erfahrungen im Regelbetrieb konnte *XERDi* kontinuierlich verbessert werden. Um auch weiterhin den Ansprüchen der Nutzer zu genügen, wird um Feedback bzgl. auftretender Fehler, Verbesserungsvorschlägen und Erweiterungen des Funktionsumfangs gebeten.

2 Inbetriebnahme

Voraussetzungen *XERDi* ist ein Javaprogramm und benötigt hierzu eine installierte Java Runtime Environment (JRE) in mind. der Version 5.0. Diese ist auf der Java-Homepage <http://www.java.com> kostenfrei beziehbar. Es wird jedoch empfohlen, mind. die Version 6.0 zu verwenden, da das Programm hiermit an Stabilität und Geschwindigkeit hinzugewinnt sowie weitere Verbesserungen der neueren Version erfährt.

Installation *XERDi* wird als selbstextrahierendes JAR-Archiv bereitgestellt. Als Installationsverzeichnis sollte ein gesondertes Verzeichnis gewählt werden. Erstellt werden die Startdatei „*XERDi.jar*“, das Handbuch als PDF-Datei, sowie weitere Dateien (Bibliotheken) im Unterverzeichnis „*lib/*“, welche zur Ausführung von *XERDi* benötigt werden.

Programmstart Die *XERDi*-Startdatei lässt sich in der Regel direkt (z.B. durch Doppelklick) ausführen. Alternativ kann der Programmstart auch über die Konsole mit dem Befehl „*java -jar XERDi.jar*“ erfolgen. Beim initialen Start wird eine Standard-Konfigurationsdatei und eine Logdatei im Unterverzeichnis „*log/*“ angelegt, in welcher Log-Nachrichten gespeichert werden. Bei jedem Bugreport sollte diese Logdatei mitgeschickt werden.

3 Programmfunktionen

Die folgende Auflistung gibt einen Überblick über die wichtigsten Funktionen und Eigenschaften von *XERDi*:

- Grafische Erstellung von (E)ER-Diagrammen unter Verwendung der Notation von ELMASRI und NAVATHE [\[EN04\]](#)
- Überführung von EER- in ER-Diagramme nach den vier in [\[EN04\]](#) genannten Möglichkeiten
- Generierung von SQL-Skripten
 - Unterstützung aller SQL-Standard-Datentypen samt Parametern und Einschränkungen
 - Unterstützung von acht SQL-Dialekten
 - Hohe Transparenz durch schrittweise Transformation
- Visuelle Hilfestellung bei der Erstellung von (E)ER-Diagrammen
- Diagrammvalidierung
- Export als Bild und Dokument
- Änderungshistorie: Rückgängigmachen und Wiederherstellen von Aktionen
- Sicherungsmechanismen für modifizierte (E)ER-Diagramme
- Automatisches Abspeichern zur Vorbeugung von Datenverlust
- Uneingeschränkte Kopier-, Einfügen und Löschfunktionen
- Minimap zum Überblick und zur Navigation
- Standardzoomfunktionen
- Mehrsprachigkeit, derzeit: Deutsch und Englisch
- Hohe Skalierbarkeit: Quasi unbeschränkte Diagrammgröße, Anzeige vieler Komponenten gleichzeitig
- Anpassung der Ressourcenbelastung: Verwendung auch auf älteren Systemen möglich

4 Bedienung

In diesem Kapitel werden alle Vorgänge zur Erstellung eines Diagramms sowie die nützlichen Zusatzfunktionen erläutert.

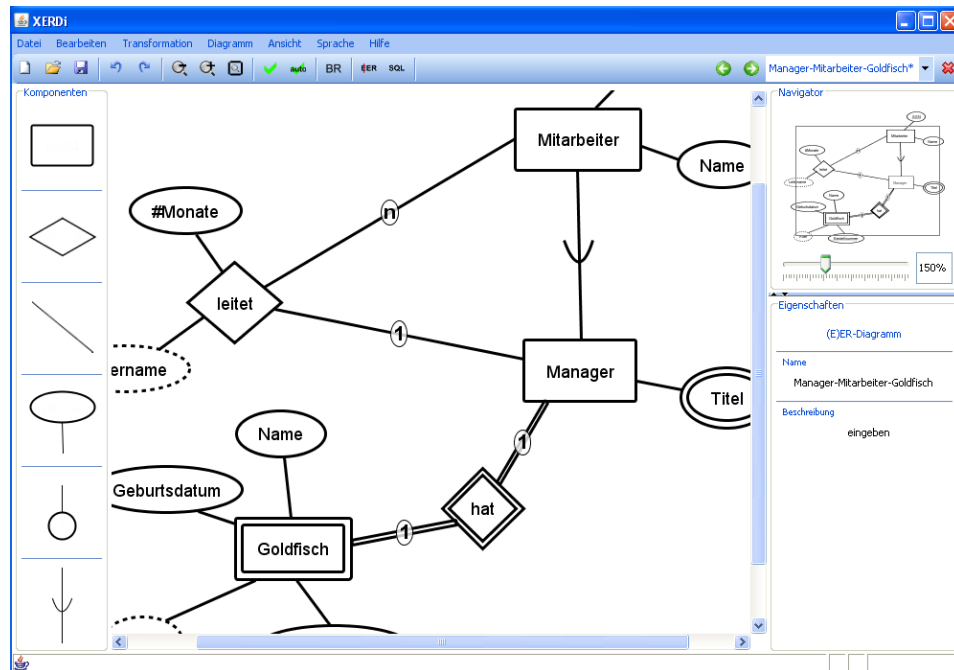


Abbildung 4.1: Benutzungsoberfläche

4.1 Quickstart

Die Funktionen zum Erstellen, Speichern, Öffnen und Schließen von Diagrammen befinden sich in der Menüleiste.

Das Hinzufügen von Komponenten wird durch einen Klick auf die gewünschte an der linken Seite angezeigte Komponente und anschließendem Platzieren auf der Diagrammfläche durchgeführt. Zum Hinzufügen von mehreren Komponenten gleichen Typs kann die STRG- bzw. CTRL-Taste gedrückt gehalten werden.

Zum Verbinden muss ein Endpunkt einer Kante auf die gewünschte Komponente gezogen werden. Es können nur Verbindungen erstellt werden, die mit dem (E)ER-Modell konform sind. Eine Kante kann auch beim Hinzufügen Komponenten verbinden, dazu muss auf die Komponenten geklickt werden.

4.2 Diagrammerstellung

In diesem Abschnitt werden die Funktionen beschrieben, welche zur Erstellung eines Diagramms benötigt werden.

4.2.1 Anlegen eines neuen Diagramms

Das Anlegen eines neuen Diagramms kann auf drei verschiedene Weisen erfolgen:

- Klicken auf den Eintrag *Neues Diagramm* in der Menüleiste unter *Datei*
- Klicken auf das entsprechende Bild in der Iconbar.
- Verwendung der Tastenkombination *STRG + n*.

4.2.2 Öffnen eines Diagramms

Das Öffnen eines Diagramms kann auf vier verschiedene Weisen erfolgen:

- Klicken auf den Eintrag *Diagramm öffnen...* in der Menüleiste unter *Datei*
- Klicken auf das entsprechende Bild in der Iconbar.
- Verwendung der Tastenkombination *STRG + o*.
- Durch Drag&Drop aus dem Dateimanager (z.B. Explorer). Hierbei können mehrere Dateien und auch Verzeichnisse selektiert werden, wodurch mehrere Diagramme auf einmal geöffnet werden.

Bei den ersten drei Varianten öffnet sich daraufhin eine Dateiauswahl, in der das gewünschte Diagramm ausgewählt werden kann.

4.2.3 Speichern von Diagrammen

Das Speichern eines Diagramms kann auf drei verschiedene Weisen erfolgen:

- Klicken auf die Einträge *Diagramm speichern*, *Diagramm speichern unter...* oder *Alle Diagramme speichern* in der Menüleiste unter *Datei*. Bei *Diagramm speichern unter...* und wenn das Diagramm noch nicht gespeichert wurde, öffnet sich ein zusätzliches Fenster, in dem der Speicherort und der Name der Datei bestimmt werden muss.
- Klicken auf das entsprechende Bild in der Iconbar.
- Verwendung der Tastenkombination *STRG + s*.

Es gibt zwei verschiedene Speicherformate. Grundsätzlich ist eine XERDi-Datei eine XML-Datei, welche mit einem Editor angezeigt werden kann. Standardmäßig wird diese Datei jedoch komprimiert, um gerade bei großen Diagrammen den Speicherbedarf zu reduzieren. Auch diese komprimierte Datei lässt sich mit einem Editor bearbeiten, indem zuvor die XERDi-Datei über einen Archivierer (z.B. 7zip) geöffnet wurde.

Beim Öffnen eines gegebenen Diagramms und anschließendem Speichern verändert sich das Speicherformat nicht. Soll dieses gewechselt werden, muss es explizit über die *Diagramm speichern unter...*-Funktion geschehen.

4.2.4 Schließen von Diagrammen

Das Schließen eines Diagramms kann auf drei verschiedene Weisen erfolgen:

- Klicken auf die Einträge *Diagramm schließen* oder *Alle Diagramme schließen* in der Menüleiste unter *Datei*. Sollte ein Diagramm modifiziert und diese Änderung nicht abgespeichert worden sein, wird eine entsprechende Information ausgegeben.
- Klicken auf das entsprechende Bild (ganz rechts) in der Iconbar.
- Verwendung der Tastenkombination *STRG + F4*.

4.2.5 Hinzufügen von Komponenten

Zum Hinzufügen einer Komponente muss die gewünschte auf der linken Seite angeklickt werden. Anschließend kann es im Diagramm platziert werden. Bei gedrückt gehaltener STRG-Taste können weitere Komponenten des gleichen Typs hinzugefügt werden. Durch einen Rechtsklick wird das Hinzufügen abgebrochen bzw. der letzte Klick zurückgenommen.

Hinzufügen einer Kante

Eine Kante wird durch zwei Klicks im Diagramm platziert. Hierbei kann es auch gleich mit einer Komponente verbunden werden, indem bei der Platzierung auf die entsprechenden Komponenten geklickt wird. Näheres hierzu siehe unter *Verbinden von Komponenten* [4.2.8](#).

Hinzufügen eines Attributes

Zu einem Attribut gehört immer eine Attributkante. Diese wird beim Hinzufügen eines Attributes ebenfalls hinzugefügt. Wenn das Attribut auf eine freie Fläche platziert wird, so ist die Attributkante an der unteren Seite angebracht. Wenn auf eine Entität, Beziehung oder anderes Attribut geklickt wird, so ist die Attributkante hieran befestigt. Das Attribut gehört damit zur angeklickten Komponente. Mit einem zweiten Klick muss nun das Attribut platziert werden.

Hinzufügen eines Generalisierungsknotens

Zu einem Generalisierungsknoten bzw. einer Generalisierung gehört immer eine Generalisierungskante. Diese wird beim Hinzufügen einer Generalisierung ebenfalls hinzugefügt. Wenn der Generalisierungsknoten auf eine freie Fläche platziert wird, so ist die Generalisierungskante an der unteren Seite angebracht. Wenn auf eine Entität geklickt wurde, so ist die Generalisierungskante hieran befestigt. Die Generalisierung bezieht sich damit auf die angeklickte Entität. Mit einem zweiten Klick muss nun der Generalisierungsknoten platziert werden.

4.2.6 Selektieren von Komponenten

Zur Selektion einer Komponente muss diese fokussiert und anschließend angeklickt werden. Es erscheint daraufhin in blauer Farbe. Es können auch mehrere Komponenten selektiert werden: Hierzu muss bei gedrückter linker Maustaste ein Rahmen um die gewünschten Komponenten gezogen werden. Durch Verwendung der rechten Maustaste können noch weitere Komponenten umrahmt werden. Einzelne Komponenten lassen sich durch Halten der STRG-Taste und linkem Mausklick hinzufügen. Um bestimmte Komponenten aus der Selektion zu entfernen, muss die Umschalt-Taste gehalten und die jeweiligen Komponenten entweder per linkem Mausklick oder durch Umrahmen mit der rechten Maustaste bestimmt werden. Bei Entitäten, Beziehungen und komplexen Attributen werden durch einen Doppelklick auch sämtliche anhängenden Attribute mit selektiert.

4.2.7 Verschieben von Komponenten

Verschoben werden können alle Komponenten, die zuvor selektiert wurden. Dazu muss auf eine selektierte Komponente die linke Maustaste gedrückt gehalten werden und anschließend an die gewünschte Position verschoben werden. Sollte dies außerhalb des Zeichnungsausschnittes liegen, wird an die entsprechende Position gescrollt. Um den derzeitigen Zustand beim Verschieben noch einsehen zu können, wird das Diagramm eingefroren und die zu verschiebenden Komponenten in schwächerer Farbe darüber gelegt. Erst nach Beendigung der Verschiebung wird das Diagramm aktualisiert; dabei werden die Kanten neu ausgerichtet. Um das Verschieben zu widerrufen, kann die Aktion rückgängig gemacht oder direkt beim Verschieben die rechte Maustaste betätigt werden.

4.2.8 Verbinden von Komponenten

Das Verbinden von Komponenten erfolgt über die Kanten. Eine Kante sollte zuerst selektiert werden, woraufhin die Kantenenden speziell gekennzeichnet werden. Das gewünschte Kantenende kann nun in die gewünschte zu verbindende Komponente verschoben werden. Zur Vereinfachung rastet der Mauszeiger am Kantenende ein. Eine Verbindung entsteht nur dann, wenn sie mit dem (E)ER-Modell konform ist; semantisch falsche

Verbindungen werden dadurch vermieden. Komponenten, mit denen keine Verbindung eingegangen werden kann, werden ausgegraut. Die erfolgreiche Verbindung wird durch die Rotfärbung des Kantenendes symbolisiert.

4.2.9 Löschen von Komponenten

Zur Löschung einer oder mehrerer Komponenten müssen diese vorher selektiert worden sein. Anschließend werden diese über den Eintrag *Entfernen* in der Menüleiste unter *Bearbeiten* oder über das Kontextmenü gelöscht. Die Betätigung der ENTF-Taste bewirkt selbiges.

4.2.10 Editieren der Eigenschaften

Die Eigenschaften der derzeit selektierten oder fokussierten Komponente werden auf der rechten Seite angezeigt, worüber sie auch editiert werden können. Über das Kontextmenü werden ebenfalls Editierungsmöglichkeiten bereitgestellt. Entitäten, Beziehungen und Attribute besitzen einen Namen, der auch im Diagramm selbst verändert werden kann. Dazu muss die Komponente zweifach angeklickt werden (kein Doppelklick!).

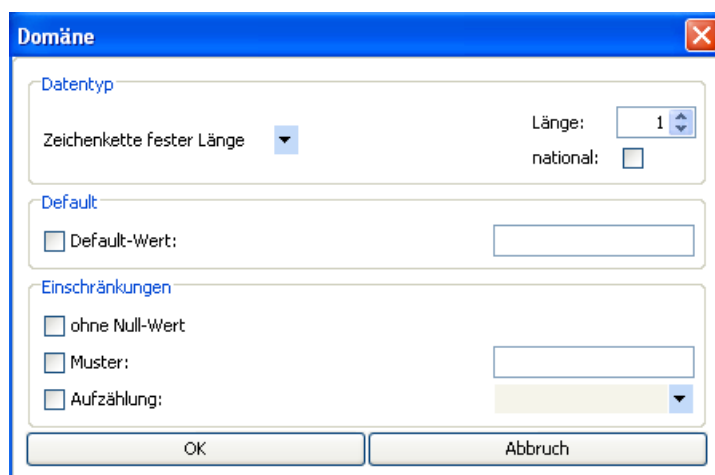
The image shows a dialog box titled 'Domäne' with a blue header bar and a red close button. It contains three main sections: 'Datentyp' (Data Type) with a dropdown menu set to 'Zeichenkette fester Länge' and a 'Länge' (Length) spinner set to '1'; 'Default' with a 'Default-Wert' (Default Value) text field; and 'Einschränkungen' (Constraints) with checkboxes for 'ohne Null-Wert' (unchecked), 'Muster' (unchecked), and 'Aufzählung' (checked). The 'Aufzählung' section also includes a text field and a dropdown menu. At the bottom are 'OK' and 'Abbruch' (Cancel) buttons.

Abbildung 4.2: Editierung der Domäne

4.2.11 Domäne eines Attributs anpassen

Die Domäne eines Attributs gibt an, welche Werte ein Attribut annehmen kann. Für die bloße grafische Diagrammerstellung hat dies keine Auswirkung. Wenn jedoch eine anschließende SQL-Generierung gewünscht ist, so muss eine entsprechende Einstellung während der Diagrammerstellung vorgenommen werden. Dazu muss das Attribut zuerst selektiert werden, damit seine Eigenschaften rechts angezeigt werden. Durch Klick auf die derzeitig eingestellte Domäne öffnet sich ein zusätzliches Fenster (siehe [Abbildung 4.2](#)).

Im oberen Teil muss der Datentyp festgelegt werden. Hierbei werden alle SQL-Standarddatentypen unterstützt. Je nach Datentyp können weitere Einstellungen vorgenommen werden. Im mittleren Teil kann ein Defaultwert bestimmt werden. Dieser wird beim späteren Hinzufügen eines Datensatzes in die Datenbank angenommen, falls für dieses Attribut kein Wert angegeben worden ist. Im unteren Teil können noch weitere Einschränkungen vorgenommen werden: Durch die Einstellung „ohne Null-Wert“ wird erzwungen, dass das Attribut immer einen Wert besitzt. Bei zahlenwertigen Datentypen können obere und untere Schranken definiert werden. Bei Zeichenketten können Muster vorgegeben werden, also z.B. eine dreistellige Zahl gefolgt von einem Bindestrich und fünfstelliger Zeichenfolge. Für näheres hierzu sei auf die SQL-Spezifikation verwiesen. Mit Ausnahme von Binärdaten und Booleschen Werten können auch Mengen festgelegt werden. Der Wert des Attributes muss dann mit einem in dieser Menge übereinstimmen. Ein Wert kann der Menge hinzugefügt werden, indem es in der Combobox geschrieben und mit der Eingabetaste bestätigt wird. Zum Entfernen eines Wertes muss es ausgewählt und anschließend die ENTF-Taste betätigt werden.

4.3 Transformation

XERDi bietet die Möglichkeit erweiterte Entity-Relationship-Diagramme in einfache zu überführen und aus einem Diagramm ein SQL-Skript zu generieren.

4.3.1 EER zu ER

Bei der Transformation von einem erweiterten Entity-Relationship-Diagramm in ein einfaches werden die Komponenten zur Generalisierung bzw. Spezialisierung durch geeignete andere Strukturen ersetzt. Hierbei sind vier verschiedene Möglichkeiten [EN04] wählbar, die als Konvertierungsoption eingestellt werden können. Die Option *auto* wählt abhängig davon, ob die Generalisierung disjunkt oder überlappend ist, entweder die Option 3 oder 4.

Für jede Generalisierung kann eine andere Konvertierungsoption gewählt werden, und die Transformation wahlweise nur für eine einzelne Generalisierung oder für alle im Diagramm befindlichen durchgeführt werden. In der Iconbar befindet sich eine Schaltfläche zur Komplettransformation. Über die Menüleiste unter *Transformation* lässt sich auch die Einzeltransformation durchführen. Die Taste F5 bewirkt die Einzeltransformation, die Taste F6 die Komplettransformation.

4.3.2 SQL-Generierung

Um sich ein SQL-Skript generieren zu lassen, ist eine Internetverbindung notwendig. Hierbei wird überprüft, ob diese Funktion in der verwendeten Version von *XERDi* und zum derzeitigen Zeitpunkt verfügbar bzw. zulässig ist. Nach einem kurzen Moment, bei größeren Diagrammen kann dies auch etwas länger dauern, wird ein neues Fenster (siehe Abbildung 4.3) geöffnet, in welchem im unteren Bereich das SQL-Skript aufgeführt ist.

Dieses kann abgespeichert oder in die Zwischenablage kopiert werden. Im linken oberen Bereich kann der gewünschte SQL-Dialekt bestimmt werden. Auf der rechten Seite können die einzelnen Generierungsschritte angewählt werden, um die Generierung nachvollziehen zu können. Unter Optionen lässt sich einstellen, ob bei fehlenden Schlüsseln automatisch welche erzeugt und ob die Bezeichner grundsätzlich in Anführungszeichen gesetzt werden sollen. Bei jeder Einstellungsveränderung muss zur wirksamen SQL-Generierung die *Aktualisieren*-Schaltfläche betätigt werden.

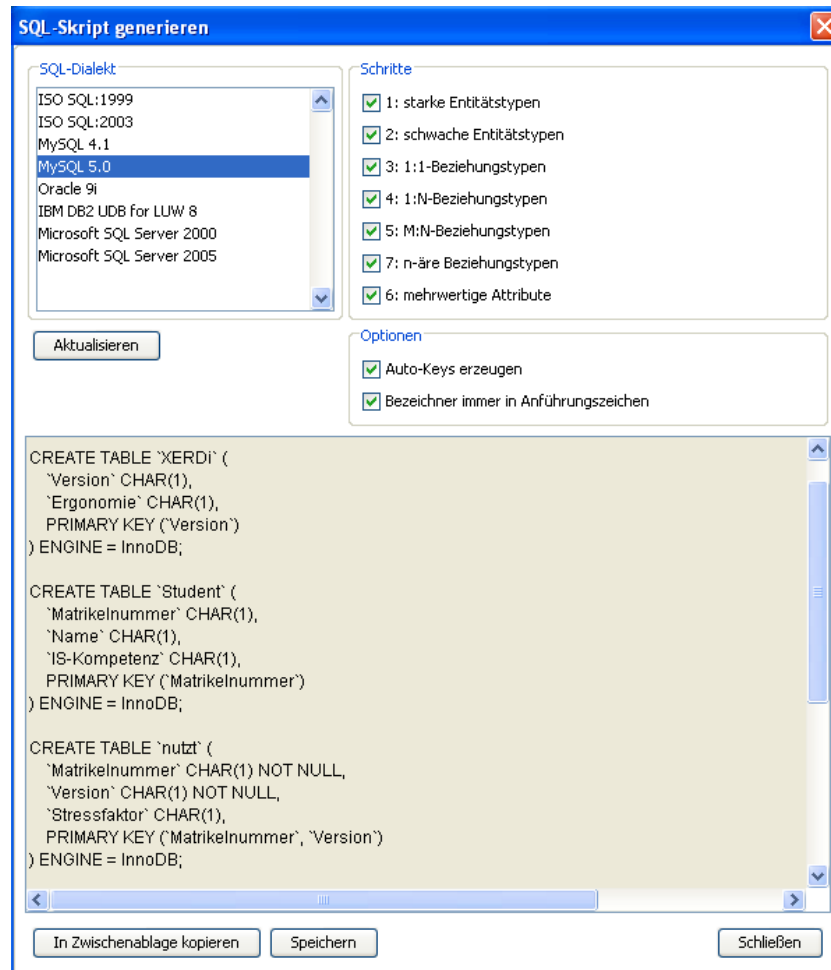


Abbildung 4.3: Generierung eines SQL-Skripts

4.4 Funktionen

In diesem Abschnitt werden weitere Funktionen aufgeführt, welche das Arbeiten mit *XERDi* angenehmer gestalten.

4.4.1 Diagrammvalidierung

XERDi bietet die Möglichkeit die strukturelle Korrektheit des Diagramms zu überprüfen. Dies kann durch die zwei Schaltflächen in der Iconbar genutzt werden.

Fehlerkategorie	Farbliche Kennzeichnung
1. Nicht sinnvoll	gelb
2. Fehlende Verbindung	orange
3. Struktur fehlerhaft	rot

Die Diagrammvalidierung überprüft das Diagramm und zeigt die ermittelten Fehler in einem separatem Fenster an. Da diese Anzeige mit der Diagrammbearbeitung synchronisiert wird, aktualisieren sich die Einträge selbstständig. Die Fehleranzeige beinhaltet bis zu 500 Einträge, wobei die Fehler nach ihrer Kategorie geordnet sind.

Die automatische Diagrammvalidierung kennzeichnet alle Komponenten entsprechend den oben aufgeführten Farben, sobald ein Fehler im Zusammenhang mit dieser Komponente auftritt. Bei Fokussierung einer fehlerbehafteten Komponente wird der schwerwiegendste Fehler in der Statusleiste angezeigt.

4.4.2 Kopieren und Einfügen

Es lassen sich alle Komponenten so kopieren wie sie zuvor selektiert wurden. Sollte hierunter ein Attribut sein, wird die entsprechende Attributkante unabhängig davon, ob sie selektiert wurde oder nicht, auch mit kopiert. Die kopierten Komponenten können nun an beliebiger Stelle – auch in einem anderen Diagramm – eingefügt werden. Der Mauszeiger gibt hierbei das Zentrum aller einzufügenden Komponenten an. Diese beiden Funktionen können entweder über das Kontextmenü und die Menüleiste unter *Bearbeiten* oder durch Verwendung der Tastenkombinationen *STRG + c* bzw. *STRG + v* genutzt werden.

Unter *Bearbeiten* kann auch eine Kopie in die Zwischenablage erfolgen. Hierbei wird aus der Selektion ein Bitmap-Bild erstellt, welches in einem beliebigen anderen Programm, welches Bilder unterstützt, eingefügt werden kann.

4.4.3 Verwaltung mehrerer Diagramme

In *XERDi* ist es möglich mehrere Diagramme gleichzeitig geöffnet zu haben. Diese können auf der rechten Seite der Iconbar angesprochen werden. Durch die grünen Pfeiltasten wird zum nächsten bzw. vorherigen Diagramm gesprungen, mit Hilfe der Combobox kann das gewünschte Diagramm direkt ausgewählt werden.

4.4.4 Änderungshistorie

Die Änderungshistorie dient dazu bestimmte Aktionen rückgängig machen zu können und auch wiederherzustellen. Hierzu sind entsprechende Schaltflächen in der Iconbar und unter *Bearbeiten* in der Menüleiste vorgesehen. Alternativ können auch die Tastenkombinationen *STRG + z* zum Rückgängigmachen und *STRG + y* zum Wiederherstellen genutzt werden.

4.4.5 Autosave

Das automatische Abspeichern dient zum Vorbeugen von Datenverlust, der z.B. dadurch entstehen kann, dass sich das Programm aufhängt. Jede Veränderung führt dabei zum Abspeichern des Diagramms. Es ist zu beachten, dass dies bei sehr großen Diagrammen zu Geschwindigkeitsbeeinträchtigungen führen kann. Diese Option lässt sich unter *Datei* in der Menüleiste an- bzw. abwählen. Sie muss bei jedem Neustart erneut gesetzt werden.

4.4.6 Export

Um das Diagramm auch außerhalb von *XERDi* verwenden zu können, wird der Export als Bild (derzeit: JPG, PNG, SVG, EPS und PDF) und als Dokument (derzeit: HTML) unterstützt. Dieser kann über *Datei⇒Diagramm exportieren* in der Menüleiste ausgeführt werden. Der Export als Dokument beinhaltet die Anzeige des Diagramms als eingebettete SVG und die Auflistung der Business Rules.

4.4.7 Zoom

Die Zoommöglichkeiten beinhalten das individuelle Einstellen und die Verwendung vordefinierter Zoomstufen sowie die Anpassung an das Diagramm, so dass dieses – so weit möglich – komplett in der Zeichnungsfläche zu sehen ist. In der Iconbar befinden sich Schaltflächen zur schrittweisen Änderung und der Diagrammanpassung. Über *Ansicht⇒Zoom* in der Menüleiste kann zusätzlich die Anwahl der vordefinierten Zoomstufen erfolgen; dort sind ebenfalls die entsprechenden Tastenkombinationen aufgeführt. Im Navigator (rechte Seite) kann über einen Slider und über eine Texteingabe auch die gewünschte Zoomstufe direkt eingegeben werden. Der Zoom ist im Intervall [26%, 400%] definiert.

4.4.8 Ausblenden von Attributen

Bei großen Diagrammen kann es dazu kommen, dass es unübersichtlich wird. Die Kernstrukturen eines Diagramms werden durch die Entitäten und Beziehungen gebildet. Diese besitzen in der Regel mehrere Attribute, welche einen erweiterten Detailgrad darstellen. Dieser kann zur besseren Übersicht ausgeblendet werden. Dabei kann die Ausblendung für alle Attribute eines Diagramms erfolgen, für einige selektierte oder für Attribute einer bestimmten Entität, Beziehung oder eines komplexen Attributs. Zum Ein- und Ausblenden von anhängenden Attributen, muss das Kontextmenü der übergeordneten Komponente aufgerufen werden. Zum Ein- und Ausblenden von selektierten oder allen Attributen ist das Kontextmenü des Diagramms zu verwenden.

Wenn die Attribute eines komplexen Attributs, welches zu einer Entität gehört, ausgeblendet wurden, können diese nicht über die Entität direkt wieder sichtbar gemacht werden. Hierzu wird die Funktion genutzt, dass durch Doppelklick auf eine Entität auch sämtliche Attribute selektiert werden. Nun können über das Kontextmenü des Diagramms die Attribute wieder sichtbar gemacht werden.

5 Kompatibilität

Grundsätzlich lässt sich *XERDi* auf allen Systemen ausführen, für die eine Java Virtual Maschine existiert. Dies sind nach Angabe von Sun (<http://www.java.com/de/download/manual.jsp>) derzeit: Windows, Linux, Solaris, Solaris SPARC und Mac OS. Da es jedoch eine Vielzahl an unterschiedlichen Systemvarianten gibt, kann es unter Umständen zu Einschränkungen kommen oder zu Fehlern führen. Die Lauffähigkeit wurde speziell unter Windows XP, BSD und Mac OS X getestet (Rechner der ARBI und HRZ). In der Tabelle 5.1 sind die bisherigen Testergebnisse aufgeführt.

System	Einschränkungen	Fehler
Windows XP	keine	keine
BSD	Die Anzeige des Handbuchs und von exportierten Diagrammen erfolgt nur, wenn der Standardbrowser geschlossen ist.	Bei Veränderungen im Diagramm (auch Fokussierung) und insbesondere beim Verschieben von Komponenten verlangsamt sich die Anzeige.
Mac OS X	Die Anzeige des Handbuchs und von exportierten Diagrammen kann nicht direkt über das Programm erfolgen; sie müssen extern aufgerufen werden.	Leopard: Die Anzeigeninitialisierung von Javafenstern jeglicher Art ist fehlerhaft. Bei größenänderbaren Fenstern, kann der Fehler behoben werden, indem die Fenstergröße verändert wird.

Tabelle 5.1: Kompatibilität

Auf Windows und Mac OS werden standardmäßig die systemeigenen Look&Feels verwendet. Durch die Parameterangabe „noNative“ kann das Metal-Look&Feel eingestellt werden. Dies lässt sich z.B. durch den Konsolenbefehl „`java -jar XERDi.jar noNative`“ bewirken. Auf allen anderen Systemen ist automatisch das Metal-Look&Feel eingestellt.

A Anhang

A.1 Tastenbelegungen und -kombinationen

Eine Übersicht über die Tastenbelegungen und -kombinationen samt ihrer Funktionen ist mit der Tabelle [A.1](#) gegeben. In folgenden Situationen besitzen einige Tasten zusätzliche Funktionen:

Rechte Maustaste:

- Hinzufügen einer Komponente: Mit einem Rechtsklick wird das Hinzufügen zurückgesetzt. Wenn bei einer Kante ein Ende bereits platziert wurde, so bewirkt der Rechtsklick die Rücknahme dieser Platzierung.
- Beim Verschieben von Komponenten: Durch einen Rechtsklick wird die Verschiebung abgebrochen.
- Selektieren mittels Selektierungsrahmens: Wird dieser mit der rechten Maustaste gezogen, werden die umrahmten Komponenten der aktuellen Selektierung hinzugefügt bzw. bei Halten der UMSCHALT-Taste aus dieser entfernt.

STRG-Taste:

- Hinzufügen einer Komponente: Wird die STRG-Taste beim Platzieren der Komponenten gehalten, können weitere Komponenten des gleichen Typs hinzugefügt werden. Wenn ein Attribut direkt mit einer Komponente verbunden wird, so bewirkt die Taste, dass noch weitere Attribute mit dieser Komponente sofort verbunden werden. Entsprechendes gilt für Generalisierungen.

UMSCHALT-Taste:

- Selektieren mittels Selektierungsrahmens: Wird die UMSCHALT-Taste beim Loslassen der Maustaste gehalten, werden die umrahmten Komponenten aus der aktuellen Selektierung entfernt.

Taste	Funktion
STRG + <n> STRG + <o> STRG + <s> STRG + UMSCHALT + <s> STRG + F4 STRG + UMSCHALT + F4 ALT + F4	Anlegen eines neuen Diagramms Öffnen eines Diagramms Speichern eines Diagramms Speichern aller Diagramme Schließen eines Diagramms Schließen aller Diagramme Programm beenden
STRG + <z> STRG + <y> STRG + <c> STRG + UMSCHALT + <c> STRG + <v> ENTF STRG + <a> STRG + <i>	Aktion rückgängig machen Aktion wiederherstellen Selektierte Komponenten kopieren Selektierte Komponenten als Bitmap-Bild in die Zwischenablage kopieren Kopierte Komponenten an der letzten Mausposition einfügen Selektierte Komponenten löschen Alles selektieren Selektierung invertieren
F5 F6 F9	Selektierte Generalisierung umwandeln Alle Generalisierungen umwandeln SQL-Skript generieren
UMSCHALT + Pfeil hoch UMSCHALT + Pfeil unten UMSCHALT + Pfeil rechts UMSCHALT + Pfeil links UMSCHALT + EINGABE	Zeichenfläche nach oben erweitern Zeichenfläche nach unten erweitern Zeichenfläche nach rechts erweitern Zeichenfläche nach links erweitern Zeichenfläche auf Diagrammgröße trimmen
STRG + <1> STRG + <2> STRG + <3> STRG + <-> STRG + <+> STRG + EINGABE	Zoom: Klein Zoom: Mittel Zoom: Groß Zoom: Verkleinern Zoom: Vergrößern Zoom: An Diagrammgröße anpassen
Rechte Maustaste	Kontextmenü

Tabelle A.1: Tastenbelegungen und -kombinationen

A.2 Datenbankentwurf und Datenmodellierung

Ziel des Datenbankentwurfs ist es, die für eine Anwendung relevanten Daten in einer strukturierten Form zu beschreiben, was mittels eines Datenmodells geschieht. Es „legt die Modellierungskonstrukte fest, mittels derer man ein computerisiertes Informationsabbild der realen Welt“ [KA99] erstellen kann.

Dieser Abschnitt ist auszugsweise aus der Arbeit von Burmeister [Bur06] entnommen worden.

Im Bereich der Datenmodellierung ist es stets von besonderer Wichtigkeit, zwischen *Schema* und *Instanz* einer Datenbasis zu unterscheiden. Während man unter Instanz den zu einem Zeitpunkt gültigen, konkreten Zustand der Datenbasis versteht, legt ein Schema nur die Struktur der Daten fest. Die tatsächlich gespeicherten Datenobjekte, die Instanz, müssen den Einschränkungen, die im Schema festgelegt sind, genügen. Ein Schema ändert sich in der Regel nur sehr selten; demgegenüber unterliegt der konkrete Zustand einer Datenbasis einer ständigen Änderung. Das Ziel der Datenmodellierung kann daher nur die Spezifizierung des Datenschemas sein. Dies geschieht im Datenbankentwurf in mehreren Phasen, die nachfolgend beschrieben werden.

A.2.1 Phasen des Datenbankentwurfs

Der Entwurf einer Datenbankapplikation sollte ebenso wie der Entwurf einer Softwareanwendung einem klar definierten Entwurfsprozess folgen. Hierbei ist ein systematischer, schrittweiser Ablauf für einen erfolgreichen Einsatz des entwickelten Systems von entscheidender Bedeutung. Abbildung A.1 veranschaulicht einen möglichen Entwurfsprozess. Die einzelnen Schritte, die dabei zu durchlaufen sind, werden im Folgenden erläutert (vgl. [EN04]).

Der Prozess des Datenbankentwurfs beginnt mit der Analyse der Anforderungen an die Applikation. In diesem Schritt gilt es, im Dialog mit dem Auftraggeber und den späteren Benutzern des Systems, den zu modellierenden Weltausschnitt (auch „Miniwelt“ genannt) abzugrenzen und die Anforderungen an die Datenhaltung und -verarbeitung zu analysieren. Das hierbei entstehende Dokument ist eine möglichst detaillierte Aufstellung dessen, was die Datenbankapplikation später zu leisten im Stande sein soll. Die Techniken und Beschreibungsmittel in dieser Phase sind denen der Softwareentwicklung recht ähnlich und werden im Rahmen dieser Arbeit nicht näher betrachtet.

Der weitere Verlauf des Entwurfsprozesses wird üblicherweise in drei Abstraktionsebenen unterteilt:

- die konzeptuelle Ebene,
- die logische Ebene,
- die physische Ebene.

Auf der konzeptuellen Ebene, die unmittelbar auf die Anforderungsanalyse folgt, wird eine strukturierte Beschreibung der zu speichernden Daten erstellt. Hierbei kommt es

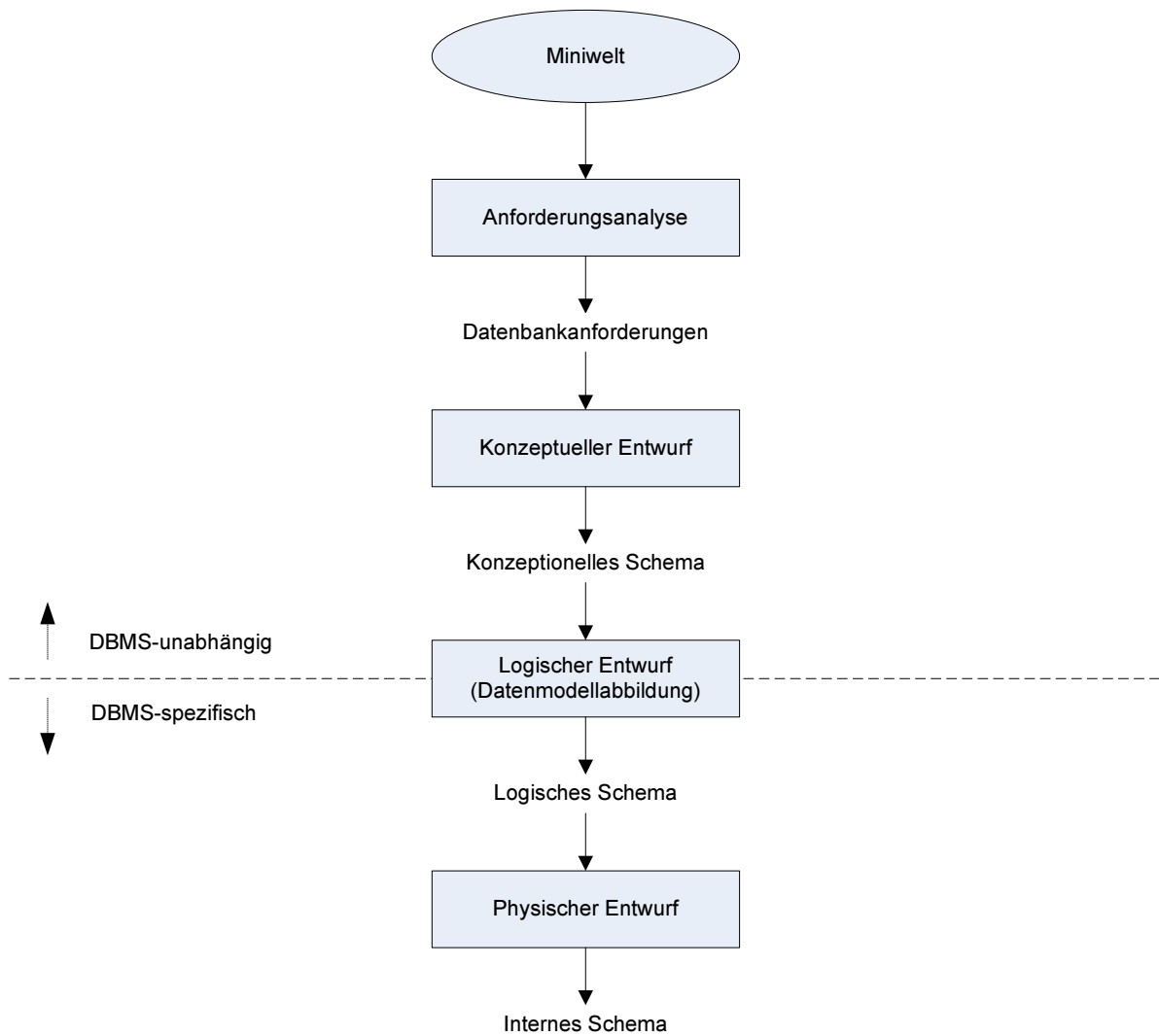


Abbildung A.1: Phasen des Datenbankentwurfs (nach [EN04])

darauf an, alle Informationen, die gespeichert werden müssen, zu identifizieren sowie Abhängigkeit und Einschränkungen zu formulieren. Ergebnis dieser Entwurfsphase ist ein konzeptionelles Schema, welches mit den Mitteln eines konzeptionellen Datenmodells erstellt wird. Die Modellierung innerhalb dieser Entwurfsphase ist sehr stark an der Anwendersicht orientiert. Wie die Daten letztendlich gespeichert werden, ist auf dieser Abstraktionsebene nicht von Bedeutung. Dies hat den Vorteil, dass unabhängig von dem später eingesetzten Datenbank-Management-Systems (DBMS) modelliert werden kann. Da die Datenmodelle des konzeptuellen Entwurfs hauptsächlich auf grafischen Notationen basieren, sind sie auch von technisch nicht versierten Personen zu verstehen und bieten daher eine gute Grundlage für die Kommunikation zwischen Datenbankentwicklern und Endbenutzern. Das mit Abstand am häufigsten verwendete konzeptionelle Datenmodell ist das Entity-Relationship-Modell. Dieses wird in Abschnitt [A.2.2](#) beschrieben.

Im nächsten Schritt, dem logischen Datenbankentwurf, findet eine Transformierung

des konzeptionellen Schemas auf ein logisches Datenmodell statt. Bei diesem Schritt sind bereits Kenntnisse über das DBMS, das eingesetzt werden soll, erforderlich. Das vorherrschende logische Datenmodell stellt das relationale Modell (s. A.2.3) dar. Ebenfalls von Bedeutung sind das objekt-relationale und das objektorientierte Modell, die jedoch nur der Vollständigkeit halber erwähnt und hier nicht weiter behandelt werden. Allenfalls von historischer Bedeutung sind das Netzwerk- sowie das hierarchische Modell. Auch diese finden hier keine weitere Betrachtung. Für den logischen Entwurf, auch als Datenmodellabbildung bezeichnet [EN04] gibt es systematische Vorgehensweisen. Ein Algorithmus für die Abbildung des Entity-Relationship-Modells auf das relationale Modell wird in Abschnitt A.2.4 beschrieben.

Am Ende des Entwurfsprozess steht der physische Entwurf, der sich mit der internen Speicherung der Daten und den Zugriffspfaden befasst. Diese Phase dient vor allem der Leistungssteigerung der Datenbankanwendung. Der physische Entwurf ist hochgradig von dem verwendeten DBMS sowie der zugrundeliegenden Hard- und Software abhängig [KA99].

A.2.2 Das Entity-Relationship-Modell

Das Entity-Relationship-Modell (ER-Modell) ist das am weitesten verbreitete Modell für den konzeptuellen Entwurf. Es wurde erstmalig 1976 von PETER CHEN vorgestellt [Che76] und seitdem von verschiedenen Seiten um zusätzliche Beschreibungsmittel erweitert. Das ER-Modell bildet den Ausgangspunkt für die Datenmodellabbildung. Die grafische Notation, die in der Literatur für ER-Diagramme verwendet wird, variiert in Teilen sehr stark. Dieser Arbeit ist die Notation von ELMASRI und NAVATHE [EN04] zu Grunde gelegt. Einen Überblick hierüber bietet Abbildung A.2.

Grundlegende Elemente

Die grundlegenden Elemente des ER-Modells sind *Entitäten* (entities), *Beziehungen* (relationships) und *Attribute* (attributes).

Entitäten repräsentieren Konzepte der realen Welt. Dies können sowohl konkrete (Gegenstände, Personen etc.) als auch abstrakte Dinge (Universität, Vorlesung etc.) sein, die über eine physische oder gedankliche Existenz verfügen. Für die Modellierung werden Entitäten in Gruppen gleichartiger Entitäten zusammengefasst, die man als Entitätstypen bezeichnet. Die Begriffe werden jedoch oft synonym verwendet. Ein Entitätstyp wird grafisch durch ein Rechteck, das den Namen des Entitätstyps beinhaltet, dargestellt.

Entitäten können untereinander in Beziehung stehen. Hierbei gilt analog zu den Entitäten, dass gleichartige Beziehungen zu Typen zusammengefasst werden. Ein Beziehungstyp wird im Diagramm durch eine Raute dargestellt, die über ungerichtete Kanten mit den Rechtecken der beteiligten Entitätstypen verbunden ist.

Sowohl Entitäten als auch Beziehungen können über Attribute verfügen. Ein Attribut stellt eine Eigenschaft dar, die eine Entität bzw. eine Beziehung näher charakterisiert. Attribute werden als Ovale dargestellt, die über Kanten mit einem Rechteck (eines Entitätstyps) bzw. einer Raute (eines Beziehungstyps) verbunden sind.

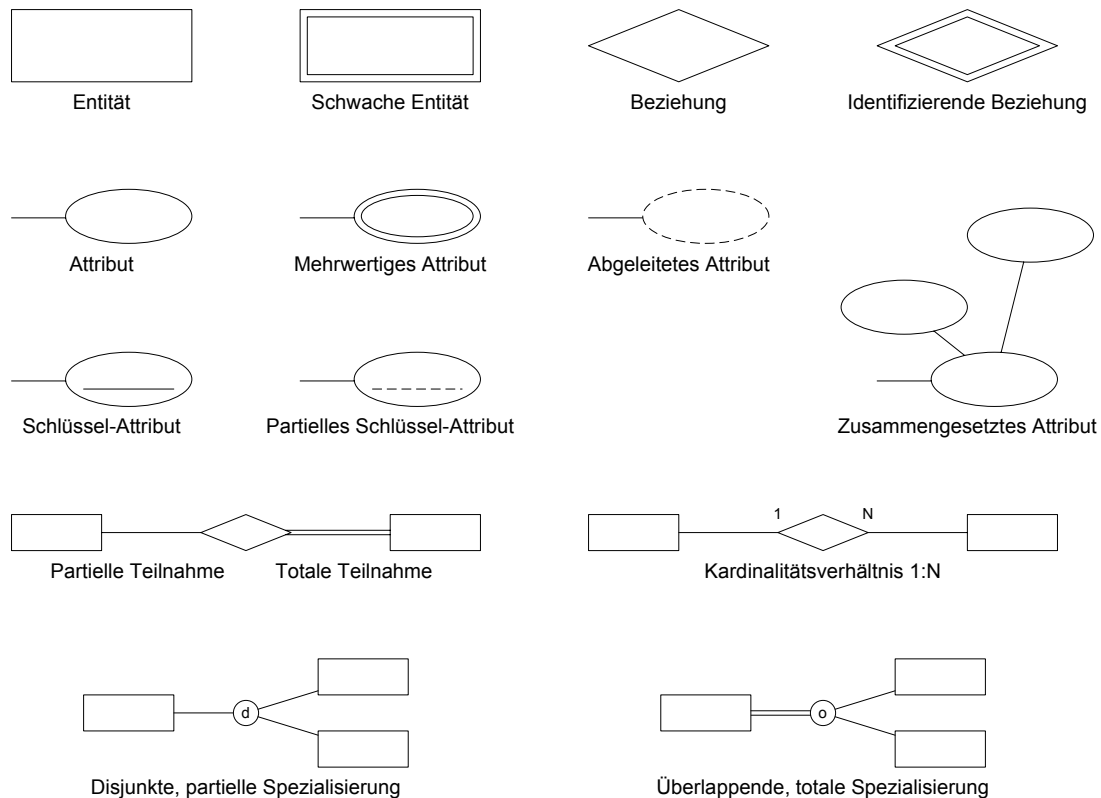


Abbildung A.2: Überblick über die Notation für (E)ER-Diagramme

Eigenschaften von Attributen

Das ER-Modell kennt unterschiedliche Arten von Attributen. Man unterscheidet zwischen *einfachen* und *zusammengesetzten*, *ein-* und *mehrwertigen* sowie *gespeicherten* und *abgeleiteten* Attributen.

Während einfache Attribute atomare (unteilbare) Werte enthalten, bestehen die Werte von zusammengesetzten Attributen aus mehreren unabhängigen Teilen. Ein Beispiel für ein zusammengesetztes Attribut ist der Name einer Person, der sich aus Vor- und Nachname zusammensetzt.

Mehrwertige Attribute unterscheiden sich von einwertigen darin, dass sie zu einem Zeitpunkt für eine Entität bzw. Beziehung mehrere Werte enthalten können. Beispielsweise kann eine Person mehrere Telefonnummern besitzen, sie besitzt jedoch in der Regel nur einen festen Wohnsitz.

Teilweise macht es Sinn, eine Entität oder Beziehung um ein Attribut zu ergänzen, dessen Wert sich aus einem anderen Attribut oder einer anderen bereits modellierten Eigenschaft ergibt. Solche Attribute bezeichnet man als abgeleitete Attribute. Das Alter

einer Person ist z.B. ein solches Attribut, da es sich aus dem Geburtsdatum der Person und dem aktuellen Datum berechnen lässt.

Neben diesen Unterscheidungen können die Attribute von Entitätstypen eine zusätzliche Eigenschaft besitzen. In der Regel verfügt jeder Entitätstyp über eine Teilmenge an Attributen, deren Werte für jede Entität des Typs eindeutig sind. Die Werte dieser Attribute identifizieren daher die zugehörige Entität eindeutig innerhalb ihres Typs. Solche Attribute werden als *Schlüssel* bezeichnet. Der Schlüssel einer Entität kann aus einem oder mehreren Attributen bestehen.

Jedem einfachen Attribut ist ein *Wertebereich* zugeordnet. Dieser legt fest, welche Werte das Attribut annehmen darf. Üblicherweise wird der Wertebereich durch einen Standard-Datentyp (z.B. Integer) spezifiziert, der durch zusätzliche Restriktionen (z.B. minimaler/maximaler Wert) eingeschränkt wird. Im (E)ER-Diagramm wird der Wertebereich in der Regel jedoch nicht dargestellt.

Eigenschaften von Beziehungstypen

Wie bereits erwähnt, dienen Beziehungstypen dazu, Assoziationen zwischen Entitätstypen zu modellieren. Die Anzahl der Entitätstypen, die an einem Beziehungstyp beteiligt sind, nennt man *Grad* des Beziehungstyps. Üblich sind binäre Beziehungen; seltener werden ternäre und höhere Beziehungsgrade verwendet. Grundsätzlich gilt, dass alle Beziehungstypen, die einen höheren Grad als zwei haben, verlustfrei in äquivalente Konstrukte überführt werden können, die nur binäre Beziehungstypen aufweisen [UJ97].

Jedem Entitätstyp, der an einem Beziehungstyp teilnimmt, lässt sich innerhalb dieser Beziehung eine *Rolle* zuordnen. Diese kann zum besseren Verständnis im Diagramm notiert werden. Meistens ergibt sich die Rolle eines an einem Beziehungstyp beteiligten Entitätstyps jedoch bereits aus dessen Namen. Daher wird die explizite Angabe der Rolle meist weggelassen. Sinn macht sie aber vor allem dann, wenn ein und derselbe Entitätstyp mehrfach an einem Beziehungstyp beteiligt ist. In diesem Fall spielt ein Entitätstyp innerhalb eines Beziehungstyps unterschiedliche Rollen. Diese sollten daher auch explizit genannt werden.

Neben der qualitativen Aussage, welche Entitätstypen an einem Beziehungstyp beteiligt sind, ist es möglich, quantitative Aussagen über die tatsächliche Anzahl von Entitäten zu machen, die miteinander in Beziehung stehen. Hierzu werden zwei Arten von Integritätsbedingungen unterschieden: die *Kardinalität* sowie die *Teilnahme einschränkung*. Über die Kardinalität legt man fest, wie viele Entitäten eines Typs maximal an einer Beziehung beteiligt sind. Man unterscheidet hierbei zwischen den Werten *1* (maximal eine Entität) und *N* bzw. *M* (beliebig viele Entitäten). Binäre Beziehungstypen lassen sich damit in drei Gruppen einteilen: 1:1-Beziehungen, 1:N- bzw. N:1-Beziehungen und M:N-Beziehungen. Bei Typen höheren Grades ergeben sich analog entsprechende Möglichkeiten.

Die zweite Form der strukturellen Einschränkung eines Beziehungstyps betrifft die minimale Anzahl der Entitäten eines Typs, die an einer Beziehung beteiligt sind. Entitätstypen können *partiell* oder *total* an einer Beziehung teilnehmen. Im ersten Fall ist die Teilnahme einer Entität an einer Beziehung nicht verpflichtend, so dass die Entität

auch ohne die Beziehung existieren kann. Im zweiten Fall ist die Teilnahme jedoch obligatorisch und es wird von jeder Entität des jeweiligen Typs verlangt, eine entsprechende Beziehung einzugehen.

Schwache Entitätstypen

In manchen Fällen ist eine Identifizierung der Entitäten eines Typs nur über eine übergeordnete Entität, mit der sie in Beziehung stehen, möglich. Man bezeichnet Entitätstypen, für die dies zutrifft, als *schwach*. Im Gegensatz zu *starken* Entitätstypen besitzen sie nur einen *partiellen* Schlüssel. Dieser identifiziert lediglich diejenigen Entitäten, die mit derselben übergeordneten Entität in Beziehung stehen. Die Beziehung wird als *identifizierend* bezeichnet.

Generalisierung und Spezialisierung

Die bisher vorgestellten Konstrukte genügen für viele Anwendungssituationen, um die Datenbasis ausreichend präzise zu beschreiben. Mit steigender Komplexität der Datenbank Anwendungen wachsen jedoch auch die Anforderungen, die man an ein konzeptionelles Datenmodell stellt. Gerade bei größeren Schemata wünscht man sich eine zusätzliche Möglichkeit der Strukturierung. Besonders vermisst man dabei Konzepte, die man unter anderem aus der objektorientierten Programmierung kennt. Diese Bedürfnisse führten zu Erweiterungen des „klassischen“ ER-Modells um die Konzepte der *Generalisierung* bzw. *Spezialisierung*. Man spricht in diesem Zusammenhang auch von dem erweiterten ER-Modell (EER-Modell).

Unter Generalisierung bzw. Spezialisierung versteht man die Erstellung hierarchischer Beziehungen zwischen Entitätstypen. Zentral sind hierbei die Begriffe der *Super-* und *Subklasse* sowie der *Vererbung*. Oftmals lassen sich die Entitäten eines Typs in mehrere Gruppen von Untertypen einteilen, die sich nur in bestimmten Eigenschaften unterscheiden. Die einzelnen Gruppen bezeichnet man als Subklassen des Entitätstyps, der selbst Superklasse genannt wird. Typischerweise unterscheiden sich die Subklassen eines Entitätstyps darin, dass sie verschiedene Attribute besitzen oder in unterschiedlichen Beziehungen stehen. Dabei gilt stets der Grundsatz der Vererbung, der besagt, dass die Subklassen alle Attribute sowie die Teilnahme in Beziehungen von der Superklasse erben. Auf Instanzebene bedeutet dies, dass alle Entitäten eines Subtyps implizit auch als Entität des Supertyps gelten und somit sowohl die Eigenschaften des Sub- als auch die des Supertyps besitzen.

Die Begriffe Generalisierung und Spezialisierung bezeichnen den Prozess der Ableitung von Super- bzw. Subklassen aus vorhandenen Entitätstypen. Dabei ist die Generalisierung, bei der bestehende Entitätstypen zu einer Superklasse verallgemeinert werden, der inverse Vorgang zur Spezialisierung, die aus einem bestehenden Entitätstyp Unterklassen ableitet. Für das Schema selbst ist jedoch nur das Ergebnis, nämlich die Existenz der Super- und Subtypen, von Bedeutung, so dass hier nicht zwischen Generalisierung und Spezialisierung unterschieden werden muss.

Zwei Einschränkungen lassen sich für Vererbungsbeziehungen formulieren. Zum einen

unterscheidet man zwischen *disjunkten* und *überlappenden* Spezialisierungen. Ist eine Spezialisierung disjunkt, kann eine Entität zu höchstens einer Subklasse seines Typs gehören. Im Gegensatz dazu erlauben überlappende Spezialisierungen die Zugehörigkeit zu mehreren Subklassen gleichzeitig. Zudem differenziert man zwischen *totalen* und *partiellen* Spezialisierungen. Totale Spezialisierungen zeichnen sich dadurch aus, dass jede Entität, die zu der Superklasse gehört, ebenfalls einem Subtyp angehören muss. Bei partiellen Spezialisierungen kann es dagegen auch Entitäten geben, die keinem speziellen Typ zugeordnet werden können.

Dokumentation

Mit der alleinigen grafischen Notation lassen sich nicht alle möglichen Strukturen modellieren. Dies gilt insbesondere dann, wenn spezielle Einschränkungen auf Beziehungen gelten müssen, damit sie bestehen, oder auch für spezielle Abhängigkeiten, z.B. zwischen Attributen. Daher ist es üblich, (E)ER-Diagrammen ergänzenden Text hinzuzufügen, welche sowohl beschreibenden Charakter besitzen als auch die o.g. Sachverhalte hinreichend spezifizieren können. *Business Rules* (BR) stellen eine verbreitete Möglichkeit dar, (E)ER-Diagramme zu dokumentieren. Es werden drei Arten von Business Rules unterschieden [ACPT00]:

- Beschreibende BRs dienen der Erläuterung der Bedeutung von Attributen, Entitäts- und Beziehungstypen für den Fall, dass der Name allein nicht aussagekräftig genug ist.
- Einschränkungen spezifizierende BRs formulieren zusätzliche Einschränkungen auf den Attributmengen oder Beziehungen von Entitäten.
- Abgeleitete Größen spezifizierende BRs dienen der Angabe einer Herleitungsvorschrift für abgeleitete Attribute.

Während das eben beschriebene (E)ER-Modell gut geeignet ist, um eine für Menschen verständliche Beschreibung der Struktur von Daten zu erstellen, basieren reale Datenbanken auf einem logischen Modell, dass unmittelbar implementiert werden kann. Der folgende Abschnitt beschäftigt sich kurz mit einem solchen Modell.

A.2.3 Das relationale Modell

Das meist eingesetzte Modell in heutigen DBMSen ist das relationale, das bereits Anfang der siebziger Jahre von EDGAR F. CODD konzipiert wurde [Cod70]. Das relationale Modell ist vergleichsweise einfach aufgebaut. Es basiert im Wesentlichen auf nur einem Strukturierungskonzept, der (mathematischen) Relation. „In dieser sehr einfachen - fast schon spartanischen - Struktur liegt aber wahrscheinlich der Erfolg der relationalen Datenbanktechnologie begründet“ [KA99].

Wie die meisten logischen Datenmodelle besteht das relationale Modell nicht nur aus den strukturellen Beschreibungsmitteln, sondern bietet darüber hinaus Basisoperatoren

zur Datenmanipulation und -anfrage. Für das relationale Modell gibt es hierfür u.a. die relationale Algebra. Für eine formale Beschreibung des relationalen Modells kann [EN04] herangezogen werden.

A.2.4 Die Datenmodellabbildung

Nachdem in den vorigen beiden Abschnitten ein konzeptionelles Datenmodell, das (E)ER-Modell, und ein logisches Modell, das relationale Modell, vorgestellt wurden, soll nun eine Abbildung zwischen diesen beiden Modellen gefunden werden. Dabei kommt es darauf an, alle Konstrukte des einen Modells in äquivalente Konstrukte des anderen Modells zu überführen, ohne hierbei einen Informationsverlust zu erleiden. Dabei genügt es an dieser Stelle die Überführung eines (E)ER-Schemas in ein relationales Schema zu betrachten, da dies die Aufgabe ist, die im logischen Datenbankentwurf gelöst werden muss. Die umgekehrte Richtung, auch als Reverse Engineering bezeichnet, stellt sich problematisch dar, da die wenigen Konzepte des relationalen Modells auf die semantisch ausdrucksstarken Konzepte des (E)ER-Modells abgebildet werden müssten¹.

Für die Abbildung zwischen den genannten Datenmodellen findet man in der Literatur unterschiedliche Algorithmen. Ein gut strukturierter und formal spezifizierter Algorithmus ist von ELMASRI und NAVATHE [EN04] vorgestellt worden. Dieser ist in acht Schritte gegliedert, welche sequentiell abgearbeitet werden können.

¹Es gibt Tools, die das Reverse Engineering von Datenbankanwendungen unterstützen. Diese erzeugen jedoch in der Regel wenig ausdrucksstarke ER-Schemata, in denen insbesondere die erweiterten Konstrukte fehlen. Oftmals ist eine manuelle Nachbearbeitung unumgänglich.

Literaturverzeichnis

- [ACPT00] ATZENI, P., S. CERI, S. PARABOSCHI, and R. TORLONE: *Database Systems: Concepts, Languages and Architectures*. McGraw-Hill Book Company, 2000.
- [Bur06] BURMEISTER, CHRISTIAN: *Automatisierung der Datenmodellabbildung zwischen (E)ER-Modell und relationalem Modell*. 2006.
- [Che76] CHEN, PETER: *The Entity-Relationship Model - Toward a Unified View of Data*, volume 1. 1976.
- [Cod70] CODD, E.: *A relational model of data for large shared data banks*, volume 13. 1970.
- [EN04] ELMASRI, RAMEZ and SHAMKANT B. NAVATHE: *Fundamentals of Database Systems*. Pearson, Addison-Wesley, 4. edition, 2004.
- [KA99] KEMPER, A. und EICKLER A.: *Datenbanksysteme*. Oldenbourg, 3. Auflage, 1999.
- [Sch06] SCHRÖDER, KIRIL: *Entwicklung einer Komponente zur grafischen Erstellung von erweiterten Entity-Relationship-Diagrammen*. 2006.
- [UJ97] ULLMAN, J.D. and WIDOM J.: *A First Course in Database Systems*. Prentice Hall, 1997.