Glossar für das Fach Informatik

In der Wissenschaft Informatik werden die Fachbegriffe nicht immer einheitlich verwendet. Zudem gibt es Unterschiede in den Darstellungsformen von Diagrammen. Das folgende Glossar soll diesem Umstand abhelfen und die für die Schulinformatik und das Landesabitur relevanten Fachbegriffe und Darstellungsformen festlegen.

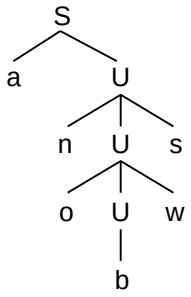
Ableitung

Als Ableitung bezeichnet man für eine gegebene Grammatik die schrittweise Bildung eines Wortes aus Terminalzeichen. Ausgehend vom Startsymbol wird bei jedem Schritt eine Produktion angewendet.

Für die unten angegebene Grammatik ist Folgendes eine Ableitung:
$S \rightarrow aU \rightarrow anUs \rightarrow anoUws \rightarrow anobws$

Ableitungsbaum

Für eine kontextfreie Grammatik kann eine Ableitung strukturiert als Ableitungsbaum dargestellt werden. Die Wurzel des Ableitungsbaumes ist das Startsymbol S, die inneren Knoten bestehen aus Nicht-Terminalen und die Blätter aus Terminalen. Die Anwendung einer Produktion wird im Ableitungsbaum durch den Produktionskopf als Elternknoten und den Produktionsrumpf als zugehörige Kindknoten dargestellt. Das obige Ableitungsbeispiel ergibt den dargestellten Ableitungsbaum.

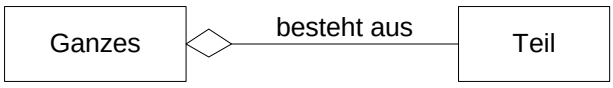


abstrakte Klasse/Methode

Mit einer abstrakten Klasse kann in einer Klassenhierarchie eine Oberklasse modelliert werden, welche abstrakte Methoden enthält, die erst in abgeleiteten Unterklassen implementiert werden. Die abstrakte Oberklasse *GeometrischeFigur* kann beispielsweise die abstrakte Methode *berechneFläche()* definieren, die jeweils in den beiden Unterklassen *Rechteck* und *Kreis* implementiert wird.

Aggregation – die *besteht aus*-Beziehung

Die Aggregation ist eine Sonderform der Assoziation zwischen zwei Klassen. Sie liegt dann vor, wenn zwischen den Objekten der beteiligten Klassen eine Beziehung existiert, die sich als „besteht aus“ oder „ist Teil von“ beschreiben lässt. In der UML-Darstellung wird die Aggregatklasse mit einer Raute versehen. Die Raute symbolisiert das Behälterobjekt, in dem die Teile gesammelt werden.



Akzeptor

Ein Akzeptor besteht aus fünf Bestandteilen:
- dem Eingabealphabet $\sum$, einer endlichen Menge von Zeichen
- der Zustandsmenge Z, einer endlichen Menge von Zuständen
- dem Startzustand z_0, einem Element aus Z
- der Menge der Endzustände E, einer Teilmenge von Z
- der Übergangsfunktion δ: $Z \times \sum \rightarrow Z$, die für Paare aus Zustand und Zeichen festlegt, in welchen Folgezustand der Akzeptor übergeht.

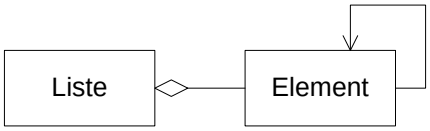
Ein Wort w wird akzeptiert, wenn ausgehend vom Startzustand z_0 schrittweise für jedes Zeichen des Wortes ein Übergang in einen Folgezustand erfolgt und nach Abarbeitung aller Zeichen ein Endzustand erreicht ist.

Assoziation – die *kennt*-Beziehung

Eine Assoziation beschreibt eine Beziehung zwischen zwei Klassen. Mit Hilfe einer gerichteten Assoziation kann dargestellt werden, dass diese Beziehung nur in einer Richtung existiert. Grafisch wird die ungerichtete Assoziation als Strecke und die gerichtete Assoziation als Pfeil dargestellt. Im Unterschied zur bidirektionalen Datenmodellierung im ER-Modell wird bei der objektorientierten Modellierung in der Regel mit gerichteten Assoziationen gearbeitet.

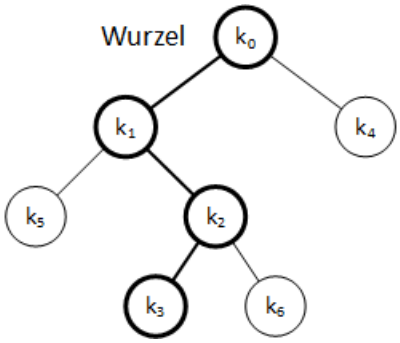
Eine Assoziation heißt **rekursiv**, wenn die beiden beteiligten Klassen gleich sind.

Beispiel: Eine lineare Liste besteht aus Elementen (Aggregation), wobei jedes Element mit Ausnahme des letzten auf das nachfolgende Element verweist (rekursive Assoziation).



Baum

Ein Baum besteht aus Knoten und Kanten. Ein einziger Knoten ist als Wurzel des Baumes dadurch ausgezeichnet, dass er keinen Elternknoten hat. Alle anderen Knoten sind Kindknoten und direkt durch eine Kante mit ihrem Elternknoten verbunden. Eine Folge k_0, k_1, ..., k_n von Knoten eines Baumes derart, dass stets k_{i+1} Kindknoten von k_i ist, wird als Pfad der Länge n bezeichnet. Knoten, die keine Kinder haben, werden als Blätter bezeichnet; alle anderen Knoten heißen innere Knoten. Die Höhe eines Baumes ist die Länge des längsten Pfades von der Wurzel zu einem Blatt. Die Tiefe eines Knotens ist die Länge eines Pfades von der Wurzel zum Knoten.



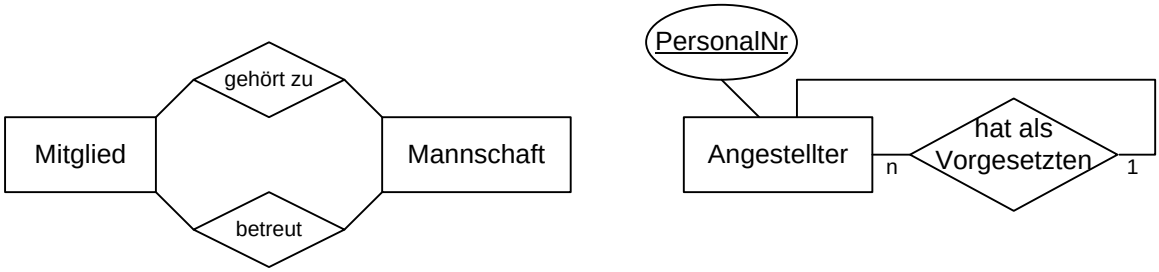
Der im Bild dargestellte Baum besteht aus 7 Knoten und 6 Kanten. Er hat die Wurzel k_0. Fett eingezeichnet ist ein Pfad mit maximaler Länge von der Wurzel bis zum Blatt k_3. Die Höhe des Baumes ist also 3 und die Tiefe des Knotens k_5 ist 2.

Beziehung und Beziehungstyp

Zwei Entitäten können in einer Beziehung stehen, z. B. die Angestellte Müller leitet die Abteilung Leichtathletik.

Ein binärer Beziehungstyp stellt eine Beziehung zwischen zwei Entitätstypen A und B dar. Die Beziehung besteht in den beiden Richtungen A → B und B → A. Daher werden Kardinalität (1:1, 1:n, n:m) und Optionalität (kann, muss) eines Beziehungstyps durch jeweils zwei Angaben beschrieben. Eigentlich müsste auch die Bezeichnung des Beziehungstyps in beiden Richtungen angegeben werden, doch üblicherweise gibt man sie nur in der Leserichtung von links nach rechts, bzw. von oben nach unten an. Im ER-Diagramm wird ein Beziehungstyp als Raute dargestellt, die den Namen des Beziehungstyps enthält.

Zwischen zwei Entitätstypen können mehrere Beziehungen bestehen. Im Beispiel gehören Mitglieder eines Vereins zu Mannschaften und jede Mannschaft wird auch von einem Vereinsmitglied betreut.



Wenn ein Entitätstyp mit sich selbst in Beziehung steht, spricht man von einer rekursiven Beziehung. Im Beispiel haben Angestellte einer Firma einen anderen Angestellten als Vorgesetzten. Bei der Abbildung ins Relationenmodell entsteht ein Namenskonflikt, weil das Attribut PersonalNr sowohl als Primärschlüssel als auch als Fremdschlüssel in der Relation erscheint. Daher wird der Fremdschlüssel gemäß der Beziehung umbenannt.

Relation: Angestellter(PersonalNr, ↑VorgesetztenPersonalNr, …)

Datenkapselung

siehe **Geheimnisprinzip**

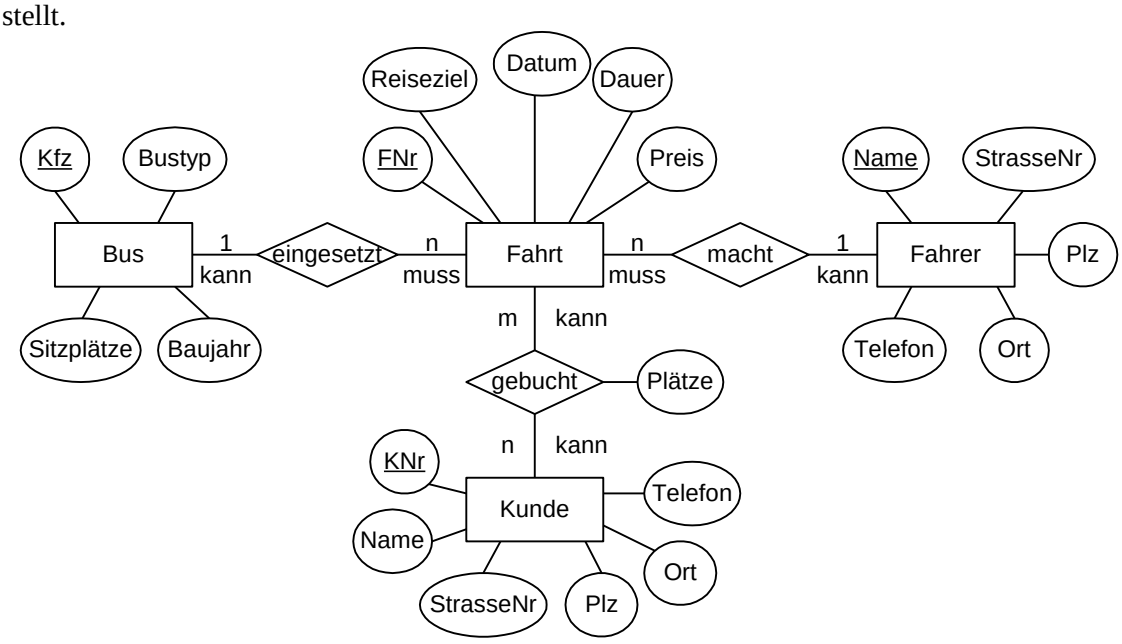
Entität und Entitätstyp

In der Datenmodellierung wird ein Objekt der realen Welt als Entität (engl.: entity) modelliert. Eine Entität kann eine Person, ein Gegenstand, ein Prozess oder auch ein nicht materielles Ding sein.

Gleichartige Entitäten bilden einen Entitätstyp (engl.: entity type) z. B. alle Angestellten, Bücher oder Reservierungen. Der Name eines Entitätstyps ist ein Substantiv im Singular. Die Eigenschaften eines Entitätstyps werden durch Attribute beschrieben. Im ER-Diagramm wird ein Entitätstyp durch ein Rechteck dargestellt, das den Namen des Entitätstyps enthält.

ER-Diagramm und ER-Modell

Die bei der datenorientierten Modellierung eines Ausschnitts der realen Welt entstehenden Entitäts- und Beziehungstypen bilden das Entity-Relationship-Modell (ER-Modell oder ERM) und werden in einem Entity-Relationship-Diagramm (ER-Diagramm oder ERD) dargestellt. Ein Entitätstyp wird durch ein Rechteck, Attribute durch Ovale und ein Beziehungstyp durch eine Raute dargestellt. Die Kardinalität eines Beziehungstyps wird im ER-Diagramm durch 1:1, 1:n bzw. n:m und die Optionalität durch „kann“ bzw. „muss“ angegeben. Exemplarisch ist nachfolgend ein ER-Diagramm dargestellt.



Fachkonzept

Das Fachkonzept ist eine zusammenfassende Darstellung eines Anwendungssystems aus fachlicher Sicht. Es besteht aus dem im Rahmen der objektorientierten Analyse entstandenen Klassendiagramm, mit Berücksichtigung aller fachlichen Aspekte des zu entwickelnden IT-Systems, ohne grafische Benutzungsoberfläche (GUI) und Datenhaltung in Datenbanken.

Geheimnisprinzip

Das Geheimnisprinzip besagt, dass die Implementierungsdetails einer Klasse verborgen werden sollen. Es wird als Datenkapselung umgesetzt, indem Attribute einer Klasse die Sichtbarkeit *private* oder *protected* erhalten und somit von außen nicht direkt zugreifbar sind. Auch der Zugriff auf Methoden kann so verhindert werden. Die Attributwerte können von außen über Methoden mit der Sichtbarkeit *public*, also über die öffentliche Schnittstelle, oder bei abgeleiteten Klassen mit Methoden der Sichtbarkeit *protected* abgefragt bzw. verändert werden.

Generalisierung → siehe auch **Vererbung**

Die Generalisierung beschreibt eine gerichtete Beziehung zwischen einer generelleren und einer oder mehreren spezielleren Klassen. Die generellere Klasse stellt eine Verallgemeinerung der spezielleren Klassen dar. Eine speziellere Klasse erbt alle Attribute der generelleren Klasse, enthält aber weitere Attribute und Methoden. Eine Generalisierung muss stets so modelliert werden, dass jedes Objekt einer spezielleren Klasse im Wortsinn auch ein Objekt der generelleren Klasse ist. Die Generalisierung ermöglicht den Aufbau von Klassenhierarchien.

Beispiel: Die Klasse Kraftfahrzeug ist eine Generalisierung der Klassen PKW und LKW, denn jeder PKW ist ein Kraftfahrzeug und jeder LKW ist ein Kraftfahrzeug.

get/set-Methode

Um ein Attribut A mit der Sichtbarkeit *private* von außerhalb der Klasse abfragen zu können, stellt man eine get-Methode *getA* zur Verfügung, die den Wert des Attributs liefert.

Um ein Attribut A mit der Sichtbarkeit *private* von außerhalb der Klasse ändern zu können, stellt man eine set-Methode *setA* zur Verfügung, die den Wert des Attributs auf den neuen Wert setzt.

Grammatik

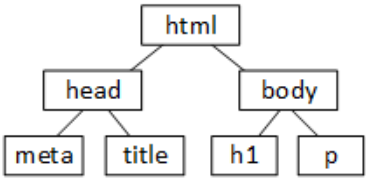
Eine Grammatik G = (N, T, P, S) besteht aus einer Menge N von Nicht-Terminalen (Variablen), einer Menge T von Terminalen, einer Menge P von Produktionen und einem Startsymbol S aus der Menge N.

Beispiel:

Nicht-Terminale	N = {S, U}
Terminale	T = {a, b, n, s, o, w}
Produktionen	P = {S → aU, U → nUs | sUn | oUw | wUo | b}
Startsymbol	S

HTML-Dokument

Die Struktur eines HTML-Dokuments wird durch aufeinander folgende und geschachtelte HTML-Elemente aufgebaut und kann durch einen Dokumentbaum dargestellt werden, dessen Wurzel das Element *html* ist.

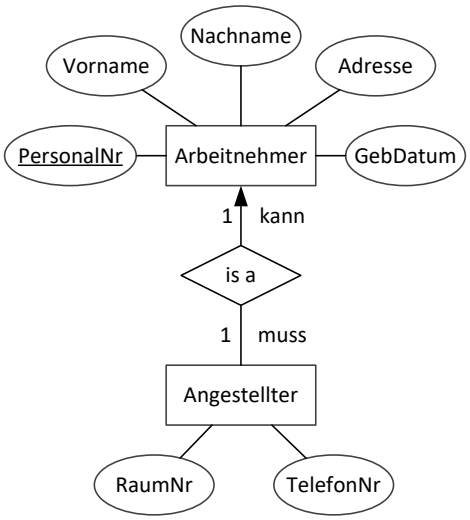


HTML-Element

Ein HTML-Element besteht in der Regel aus einem Start-Tag, dem Inhalt und dem zugehörigen End-Tag. Der Inhalt kann, je nach HTML-Element, aus reinem Text ohne HTML-Elemente oder weiteren HTML-Elementen bestehen. Beispielsweise hat das ol-Element … das Start-Tag <ol>, den Inhalt … und das End-Tag .

is a-Beziehung

Das Konzept der Generalisierung bzw. Vererbung kann auch bei der Datenmodellierung verwendet werden. Dabei ist ein Entitätstyp Untertyp eines übergeordneten Obertyps und hat normalerweise zusätzliche Attribute. Da jede is a-Beziehung die Kardinalität 1:1 und die Optionalität muss:kann hat, wird auf diese Angaben normalerweise verzichtet.



Beispiel:
In einer Firma werden Angestellte und Arbeiter als Arbeitnehmer beschäftigt. Dabei ist ein (engl. is a) Angestellter (Arbeiter) stets auch ein Arbeitnehmer. Im Relationenmodell wird ein Angestellter durch die Personalnummer identifiziert:

Angestellter(↑PersonalNr, RaumNr, TelefonNr)

Join

Ein Join verbindet zwei Relationen R und S zu einer neuen Relation R ⋈ S. Zuerst wird das Kreuzprodukt der beiden Relationen gebildet und dann eine Selektion über einen Vergleich zweier Attribute A und B der beiden Relationen durchgeführt.

Beim *Equi Join* müssen die beiden Attribute A und B den gleichen Wert haben. Als Schreibweise wird benutzt: $R \bowtie_{A = B} S$

Der *Natural Join* ist ein Equi Join, bei dem die beiden Attribute die gleiche Bezeichnung haben. Er kommt relativ oft vor, weil bei der Abbildung eines ER-Diagramms in das Relationenmodell die Beziehungen mittels Fremdschlüsseln realisiert werden, die die gleichen Bezeichnungen wie die Primärschlüssel haben. Als Schreibweise wird R ⋈ S benutzt. Gibt es in den Relationen R und S mehrere Attribute mit gleicher Bezeichnung, so müssen beim Natural Join paarweise die Attributwerte in den gleichbezeichneten Attributen übereinstimmen. Von den doppelt vorkommenden Spalten erhält der Natural Join jeweils nur eine.

Beispiel: Im unten angegebenen Relationenmodell ist *Fahrer* ⋈ *Fahrt* der Natural Join der beiden Relationen Fahrer und Fahrt über das gemeinsame Attribut Name, das in der Relation Fahrer Primärschlüssel und in der Relation Fahrt Fremdschlüssel ist.

Kardinalität

Die Kardinalität beschreibt den Grad einer Beziehung in einer relationalen Datenbank zwischen zwei Entitätstypen. Es gibt die drei Kardinalitäten 1:1, 1:n und n:m. Der manchmal benutzte Begriff Komplexität ist der Zeit- und Platzkomplexität vorbehalten.

Klasse

Eine Klasse ist die Beschreibung der Attribute (Eigenschaften) und Methoden von Objekten. Grafisch werden Klassen durch Rechtecke mit Namen, Attributen und Methoden dargestellt. Das Wort „Objektklasse“ ist eine irreführende Vermischung von Objekt und Klasse.

Klassendiagramm

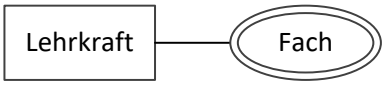
Ein Klassendiagramm stellt die Klassen und Beziehungen (Assoziation, Aggregation, Generalisierung/Vererbung) zwischen Klassen grafisch dar.

Komposition

Die Komposition ist eine Sonderform der Aggregation. Sie drückt aus, dass die Teile von der Existenz des Ganzen abhängig sind. Da meist keine klare Unterscheidung zwischen Komposition und Aggregation möglich ist, wird auf die Komposition verzichtet.

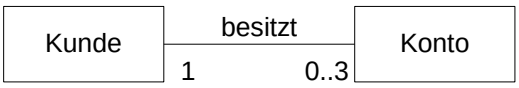
mehrwertiges Attribut

Im ER-Diagramm werden mehrwertige Attribute wie z. B. die Fächer einer Lehrkraft in einem Oval mit Doppellinie dargestellt. Bei der Überführung in das Relationenmodell wird zur Einhaltung der 1. Normalform eine eigene Relation für die Fächer benötigt.



Multiplizität

Die Darstellung von Assoziationen kann man durch Angabe von Multiplizitäten verfeinern. Dabei wird in der Minimum..Maximum-Schreibweise angegeben, wie viele Objekte der einen Klasse mit wie vielen Objekten der anderen Klasse in Beziehung stehen können.



Im Bild ist die Assoziation *besitzt* zwischen den Klassen Kunde und Konto modelliert. Die Multiplizität 0..3 gibt an, dass ein Kunde 0 bis 3 Konten besitzen kann; die Multiplizität 1 gibt an, dass ein Konto genau einem Kunden gehört.

Nicht-Terminal

Nicht-Terminale, auch Variablen genannt, sind Bestandteil einer Grammatik. Produktionen einer Grammatik legen fest, wie Nicht-Terminale durch eine Folge von Nicht-Terminalen und Terminalen in einem Ableitungsschritt ersetzt werden können.

Objekt

Ein Objekt ist ein Exemplar einer Klasse.

Objektorientierte Analyse

Phase des Entwurfsprozesses, in der das Fachkonzept entwickelt wird.

Objektorientierter Entwurf

Phase des Entwurfsprozesses, in der für das Fachkonzept eine Softwarearchitektur entwickelt wird, die die Benutzungsoberfläche und Datenhaltung mit einbezieht.

Optionalität

Beziehungstypen im ER-Diagramm lassen sich durch Kardinalität und Optionalität charakterisieren. Bei einem optionalen Beziehungstyp müssen Entitäten des einen Entitätstyps nicht mit den Entitäten des anderen Entitätstyps in Beziehung stehen. Optionale Beziehungstypen werden im ER-Diagramm durch das Wort „kann“ gekennzeichnet, obligatorische Beziehungstypen durch das Wort „muss“.

Primärschlüssel

Ein Primärschlüssel besteht aus einem oder mehreren Attributen, durch die jede Entität eines Entitätstyps eindeutig identifiziert wird. Im ER-Diagramm und Relationenmodell werden Primärschlüsselattribute unterstrichen.

Produktion

Jede Produktion einer Grammatik besteht aus einem Produktionskopf, einem Ableitungspfeil und einem Produktionsrumpf. Eine Produktion gibt an, durch welche Terminale und Nicht-Terminale der Produktionskopf ersetzt werden kann.

Beispiel: Die Produktion $U \rightarrow nUs$ legt fest, dass das Nicht-Terminal U durch nUs ersetzt werden kann.

Registermaschine

Eine Registermaschine besteht aus unendlich vielen Registern R1, R2, R3, ... in denen natürliche Zahlen gespeichert werden. Die Eingabe steht in den ersten k Registern, alle weiteren Register enthalten anfangs die Zahl 0. Sie verfügt zusätzlich über einen Akkumulator, der das Ergebnis des jeweils letzten Rechen- oder Ladebefehls enthält. Die Arbeitsweise wird durch ein Programm angegeben. Dieses setzt sich aus einer Folge von einzelnen Befehlen zusammen, die zeilenweise untereinandergeschrieben werden. Programmzeilen kann man mit einer Marke versehen, um Sprünge dorthin ausführen zu können.

Die Registermaschine hat die Befehle: LOAD, STORE, ADD, SUB, MUL, DIV, GOTO, JZERO, JNZERO und END. Bei den Sprungbefehlen wird als Operand eine Marke M angegeben. Bei den arithmetischen Befehlen und den Speicherbefehlen sind drei Arten von Operanden x zulässig:

	Operand	**Bedeutung**	**Beispiel**
Konstanten	#i	der Operand ist die Zahl i	LOAD #7
direkte Adressierung	i	der Operand ist das Register i	ADD 4
indirekte Adressierung	*i	der Operand ist das Register, dessen Nummer im Register i steht	STORE *3

In der folgenden Befehlsübersicht bedeutet x den jeweiligen Operanden:

Befehl	**Bedeutung**
LOAD x	Lädt x in den Akkumulator.
STORE x	Speichert den Wert des Akkumulators in x. Dabei darf x keine Konstante #i sein.
ADD x	Addiert x zum Akkumulator.
SUB x	Subtrahiert x vom Akkumulator. Ist der Wert von x größer als der des Akkumulators, ist das Ergebnis 0.
MUL x	Multipliziert x mit dem Akkumulator.
DIV x	Dividiert den Akkumulator durch x. Für x = 0 wird das Programm beendet.
GOTO M	Ein unbedingter Sprung zur Marke M.
JZERO M	Falls der Akkumulator den Wert 0 hat, erfolgt ein Sprung zur Marke M.
JNZERO M	Falls der Akkumulator einen Wert größer 0 hat, erfolgt ein Sprung zur Marke M.
END	Das Programm wird beendet.

Regulärer Ausdruck

Ein regulärer Ausdruck ist eine Zeichenkette, mit der eine reguläre Sprache in Form eines Musters beschrieben wird. Das Muster besteht aus Terminalen der Sprache und Metazeichen zur Konstruktion eines regulären Ausdrucks.

Metazeichen	Bedeutung
.	Der Punkt ist Platzhalter für ein beliebiges Zeichen außer für neue Zeile: `\n`
\	Der Backslash hebt die besondere Bedeutung von Metazeichen auf, um diese als Text suchen zu können, bzw. macht aus Buchstaben Steuerzeichen.
...|...	Stellt Alternativen für das Suchmuster. Die erste auftretende Alternative im String wird gefunden.
[...]	Die in den eckigen Klammern stehenden Zeichen werden als Alternative verwendet. Es können Bereiche angegeben werden, z. B.: [a-p], [3-8]. [^...] negiert die Klasse. Die Zeichenklasse steht für ein Zeichen, kann aber mit Wiederholungszeichen (*, ?, +, {n,m}) vervielfältigt werden.
(...)	Dient der Gruppierung von Suchmustern. Das gefundene Muster wird in ein Subpattern für spätere Verwendung gespeichert.
?	Erkennt das vorhergehende Element 0- oder 1-mal.
*	Erkennt das vorhergehende Element 0-, 1- oder n-mal.
+	Erkennt das vorhergehende Element 1- oder n-mal.
{n,m}	Erkennt das vorhergehende Element n-mal bis höchstens m-mal. ',m' kann entfallen, dann erkennt {n} das Element n-mal und {n,} beliebig oft, aber mindestens n-mal.

Der reguläre Ausdruck (a|b)(ab)+b? beschreibt Wörter aus den Terminalen a und b, die mit einem a oder b beginnen, worauf ein oder mehrere ab-Paare und zum Schluss ein optionales b folgen.

Mit dem regulären Ausdruck [A-Z]{1,3}-[A-Z]{1,2} [1-9][0-9]{0,3} können übliche Autokennzeichen beschrieben werden.

Relation

Eine Relation besteht aus Attributen, die die gemeinsamen Eigenschaften der Entitäten repräsentieren, und aus Tupeln konkreter Attributwerte, welche im Datenbankbereich als Datensatz bezeichnet werden. Anschaulich wird eine Relation als Tabelle dargestellt. Das Beispiel zeigt die Relation *Kurs* mit den Attributen *Kurs-Nr, Thema* und *Kurshalbjahr* sowie vier Tupeln.

Kurs

Kurs-Nr	**Thema**	**Kurshalbjahr**
13	Analysis 3	Q4
2	Short Stories	Q1
38	Datenbanken	Q2
19	Antihelden	E2

Relationenalgebra

Die Relationenalgebra definiert Operationen, die auf Relationen angewendet werden können. Als Mengenoperationen stehen *Vereinigung*, *Schnitt* und *Mengendifferenz* sowie das *Kreuzprodukt* zur Verfügung. Datenbankspezifische Operationen sind die *Selektion* $\sigma_{\text{Bedingung}}$ (R) zur Auswahl von Tupeln (Zeilen) einer Relation R gemäß einer Bedingung, die *Projektion* $\pi_{\text{Attribute}}$ (R) zur Auswahl von Attributen (Spalten), die *Umbenennung* $\rho_{\text{alt} \rightarrow \text{neu}}$(R) von Attributen und der *Join* $R \bowtie S$ zur Verknüpfung zweier Relationen R und S.

Relationenmodell

Ein Relationenmodell besteht aus Relationen, wobei jede Relation durch ihren Namen und die in Klammern gesetzten Attribute angegeben wird. Entitäts- und Beziehungstypen eines Entity-Relationship-Modells werden mittels Abbildungsregeln in ein Relationenmodell transformiert, welches Grundlage für die Realisierung einer Datenmodellierung in einem relationalen Datenbanksystem ist. Das oben angegebene ER-Modell führt zu folgendem Relationenmodell:

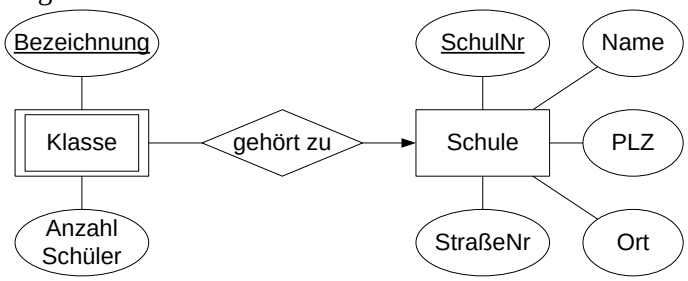
Bus (Kfz, Bustyp, Sitzplätze, Baujahr)
Fahrer(Name, StrasseNr, PLZ, Ort, Telefon)
Fahrt (FNr, Reiseziel, Datum, Dauer, Preis, ↑Kfz, ↑Name)
Kunde(KNr, Name, StrasseNr, PLZ, Ort, Telefon)
Buchung(↑FNr, ↑KNr, Plätze)

Primärschlüssel werden unterstrichen und die Beziehungstypen herstellenden Fremdschlüssel mit einem vorangestellten Pfeil ↑ gekennzeichnet.

Schwacher Entitätstyp

Einen Entitätstyp, dessen Entitäten nicht durch eigene Attribute, sondern nur durch eine zusätzliche Beziehung zu einer Entität eines übergeordneten Entitätstyps identifiziert werden können, nennt man einen schwachen Entitätstyp. Schwache Entitätstypen werden durch Rechtecke mit einer Doppellinie dargestellt und ein Pfeil zeigt zum übergeordneten Entitätstyp.

Das folgende ER-Diagramm enthält den schwachen Entitätstyp Klasse. Es gibt mehrere Schulen mit Klassen der Bezeichnung 7f.



Als Relation ergibt sich: Klasse(Bezeichnung, ↑SchulNr, AnzahlSchüler)

Sichtbarkeit

Über die Sichtbarkeit legt man fest, wer auf Klassen, Attribute und Methoden Zugriff hat. Abiturrelevant sind folgende Stufen der Sichtbarkeit: *private* (-), *protected* (#) und *public* (+). Mit Hilfe der Sichtbarkeit realisiert man das Geheimnisprinzip.

Spezialisierung

siehe **Generalisierung**, **Vererbung**

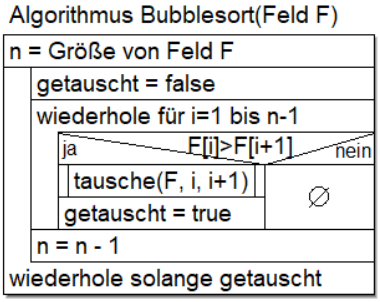
SQL – Structured Query Language

SQL ist eine Datenbanksprache mit der Daten in einem relationalen Datenbanksystem abgefragt, bearbeitet, definiert und gesichert werden können. Abfragen mittels SELECT…FROM…WHERE können neben arithmetischen, logischen und Vergleichsoperatoren auch die Operatoren COUNT, MIN, MAX, SUM, AVG, DISTINCT, AS, LIKE, BETWEEN, IN, IS NULL und die Klauseln ORDER BY, GROUP BY und HAVING enthalten. Damit lassen sich die grundlegenden Operationen Selektion, Projektion und Join der Relationenalgebra umsetzen. Der *Natural Join* kann mit NATURAL JOIN, der *Equi Join* mit JOIN…ON implementiert werden.

Weiterführende Sprachelemente ermöglichen geschachtelte SELECT-Anweisungen, das Erstellen von Tabellen mit CREATE, das Einfügen, Ändern und Löschen von Daten mit INSERT, UPDATE und DELETE, sowie die Sicherung mit GRANT und REVOKE.

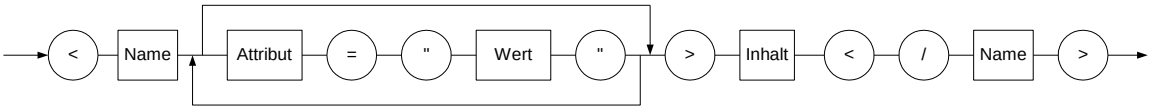
Struktogramm

Ein Struktogramm stellt einen Algorithmus unabhängig von einer konkreten Programmiersprache mit Hilfe elementarer Strukturblöcke für Sequenz, Auswahl, Wiederholung und Blockaufruf grafisch dar.



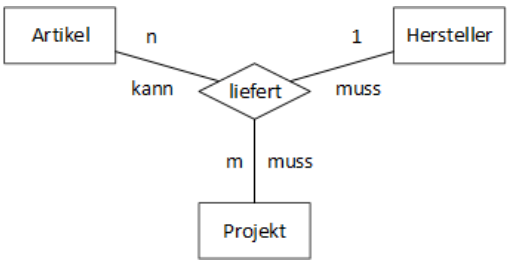
Syntaxdiagramm

Mit Syntaxdiagrammen kann man kann man die Syntax von formalen Sprachen grafisch darstellen. Terminale werden in Ovalen, Nicht-Terminale in Rechtecken dargestellt. Das folgende Syntaxdiagramm zeigt die Syntax eine HTML-Elements.



Ternäre Beziehungstypen

Ein ternärer Beziehungstyp stellt die Beziehung zwischen drei Entitätstypen dar. Bei der Abbildung ins Relationenmodell entsteht eine Relation, welche die drei Primärschlüssel der beteiligten Entitätstypen sowie vorhandene Beziehungsattribute enthält.



Umbenennung

Die Umbenennung ist eine Operation der Relationenalgebra, mit der man Attribute einer Relation umbenennen kann. Sie wird mit dem griechischen Buchstaben ρ (rho, engl. rename) bezeichnet.

Mit $\rho_{A\to X,\, B\to Y,\, D\to Z}(R)$ werden in der Relation R(A, B, C, D, E, F) die Attribute A, B und D in X, Y und Z umbenannt. Es entsteht so die Relation R(X, Y, C, Z, E, F).

Die Umbenennung wird zur Lösung von Namenskonflikten bei Join-Operationen benötigt, z. B. Ausschluss von Attributen beim *Natural Join*, Joins bei rekursiven Beziehungen oder gleichbezeichnete Attribute mit unterschiedlicher Bedeutung.

UML

Die Unified Modeling Language (UML, dt.: vereinheitlichte Modellierungssprache), ist eine standardisierte Beschreibungssprache, um Strukturen und Abläufe in objektorientierten Softwaresystemen darzustellen. Für den Informatikunterricht sind besonders das Klassendiagramm und das Zustandsdiagramm (theoretische Informatik) von Bedeutung.

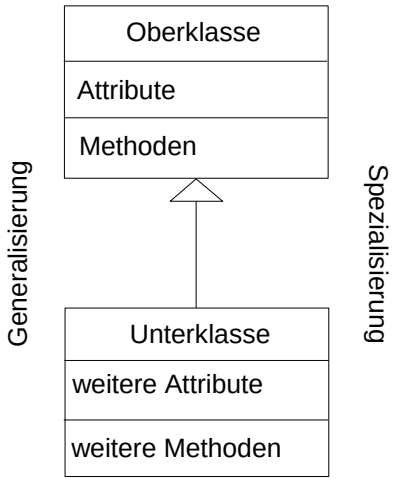
Vererbung – die ist-Beziehung

In der objektorientierten Modellierung kann eine Klasse von einer anderen Klasse erben. Die erbende Unterklasse wird von der Oberklasse abgeleitet. Sie hat Zugriff auf die geerbten Attribute und Methoden der Oberklasse, hat aber weitere Attribute und Methoden.

Die Vererbung wird mit einem geschlossenen Dreieckspfeil von der abgeleiteten Klasse zur Oberklasse dargestellt. Jedes Objekt der abgeleiteten Klasse muss im Wortsinn auch ein Objekt der Oberklasse (ist-Beziehung) sein.

Die Umkehrung der Generalisierung ist die Spezialisierung, welche durch Vererbung realisiert wird.

Beispiel: Ein Motorrad ist ein spezielles Kraftfahrzeug.



zusammengesetztes Attribut

Im ER-Diagramm kann man ein zusammengesetztes Attribut wie z. B. Adresse verwenden. Beim Überführen in das Relationenmodell muss es in die einzelnen Attribute StraßeNr, PLZ und Ort aufgeteilt werden, damit die 1. Normalform eingehalten wird.

Stand: 20.06.2020