Es gibt verschiedene Umsetzungen von Datenbankmanagementsystemen:

- RDBMS: relationale DBMS,
- ODBMS: objektorientierte DBMS,
- XDBMS: XML DBMS.
- ...

Relationale sind die üblichsten: **MySQL**, **PostgreSQL**, **Oracle**, **Sybase**, Microsoft Access, ...

Beispiele für ODBMS findet man unter **odbms.org**.

Beispiele für XDBMS sind **Sedna**, **BaseX**, ...

Wir verwenden das RDBMS **SQLite** mit graphischer Oberfläche **SQLiteCC** oder **SQLite Expert**. Möglicherweise nutzen wir auch noch die Einbettung in Python.

Datenbankentwurf

Entwicklungsprozess
Anforderungsanalyse & Miniwelt

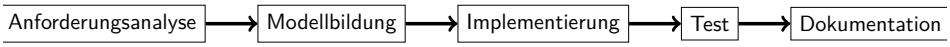
Wollen DB entwickeln. Etwa für Comic-Sammlung, aus der Freunde ausleihen dürfen. Was ist dazu zu tun? Wie kommt man zu einer laufenden Anwendung?

Entwicklungsprozess

Kennen wir aus der Programmierungsvorlesung:

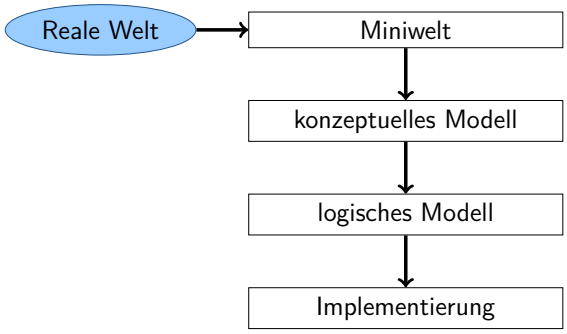
Software–Entwicklungsprozess	Datenbankentwurf
1. Problemformulierung, 2. Algorithmus, 3. Implementierung, 4. Test und 5. Dokumentation.	1. Anforderungsanalyse, 2. Design oder Modellbildung, 3. Implementierung, 4. Test und 5. Dokumentation.

Häufig wiederholt durchführen für mehr Features und weniger Fehler: v1.0, v1.1, . . . , v2.0, . . .



- ‚*Anforderungsanalyse*‘: Worum geht es eigentlich?
- ‚*Modellbildung*‘: Abstraktion der wahren Welt auf allgemeine Strukturen.
  ↪ Entity-Relationship-Modelle, Relationen/Tabellen
- ‚*Implementierung*‘: Überführung des Modells auf den Computer.
  ↪ SQL
- ‚*Test*‘, ‚*Dokumentation*‘: stellen sicher, dass wir das bekommen, was wir wollen und das auch neue Benutzer damit umgehen werden können.

Noch eine etwas andere Sicht:



Gewünschte Anwendung lebt in der realen Welt, verwendet aber nur kleinen Ausschnitt, Miniwelt.

Miniwelt wird konzeptuell modelliert, bspw. mittels Entity-Relationship-Diagrammen oder UML-Diagrammen.

Um (relationalen) Datenbank näher zu kommen, wird aus ER-Diagrammen ein logisches Modell über Tabellen abgeleitet und optimiert.

Letztlich Implementierung in SQL.

Anforderungsanalyse & Miniwelt

Erster (und wichtigster) Schritt:

Alle Anforderungen an die DB erfassen.

Vom *Endanwender* erfahren

1. um welche Daten handelt es sich, und
2. was soll mit den Daten angefangen werden.

Dazu

- Interviews mit allen Beteiligten führen, um Anwendungsfälle zu erfassen.
- Interviewtexte analysieren bzgl. Daten und Zusammenhängen zwischen Daten.
- Mit grafischer Modellierung (Entity-Relationship-Diagramme) erneut Interviews führen und verfeinern.

Interviews bewegen sich auf der konzeptuellen Ebene und beschreiben die *‚Miniwelt‘*. Ausschnitt der realen Welt, der für fragliche Anwendung von Interesse ist.

Miniwelt besteht aus gewissen Objekten mit Attributen und Beziehungen (mit Attributen) untereinander.

Objekte, Attribute und Beziehungen müssen in DB abgebildet werden.

Miniwelt ‚Comicsammlung‘ (vielleicht unvollständig):

> Ich habe eine Comic-Sammlung mit relativ vielen verschiedenen Comics. Etwa klassische Bildergeschichten wie ‚Tim&Struppi‘, moderne Geschichten wie ‚Sin City‘. Aber auch Einzelbände wie ‚Logicomix‘. Doppelte Hefte sortiere ich sofort aus. Natürlich will ich soviel Information wie möglich über den einzelnen Comic und zugehörige Serie.
>
> Seit ich die Sammlung habe und einzelne Comics verleihe, habe ich auch viele Freunde. Mein Wunsch ist es, meine Sammlung im Computer abzuspeichern. Ich möchte wissen, welche Comics ich alle habe. Bspw. welche von Tim&Struppi. Auch meine Freundinnen will ich abspeichern. Ich möchte wissen, wer welche Comics ausgeliehen hat und ob sie schon zurückgegeben wurden. Und wie ich den Ausleiher kontaktieren kann.

Miniwelt ‚Hochschule' (sicherlich unvollständig):

> An einer Hochschule bewegen sich Studierende und Dozenten und weitere Mitarbeiter. Sie kommen zusammen in Vorlesungen und Prüfungen (und in der Mensa). Betrachten wir hier nur den Aspekt der Lehre. Dann besitzt jeder Studierende eine Matrikelnummer, die ihn *eindeutig* während seines gesamten Studiums an der Hochschule begleitet. Sie hat einen Namen und einen Studiengang, den sie studiert. Und darüber hinaus hat er eine aktuelle Anzahl von Semestern, die er schon studiert.
>
> Ein Dozent macht (neben Forschung) Vorlesungen in bestimmten Fachrichtungen und vorgegebenen Umfang zu hoffentlich interessanten Themen. Auch ein Dozent hat einen Namen und für die Hochschule eine eindeutige Personalnummer. Die Vorlesungen haben alle eine Titel und werden in einem bestimmten Semester gehalten. An den Vorlesungen nehmen Studierende teil und sie schreiben auch Prüfungen zu diesen Veranstaltungen mit hoffentlich positiven Ergebnissen.

Herausarbeiten der Objekte, Beziehungen und zugehörigen Attribute auf folgende Art und Weise:

- ▷ Analysiere alle Nomen im Text. Diese können als Objekte oder Attribute dienen.
- ▷ Analysiere alle Verben im Text. Diese können als Beziehungen dienen.
- ▷ Analysiere alle Adjektive im Text. Diese können als Attribute dienen.

Aktion: Durchspielen für Hochschulwelt.

- Studierende = (Matrikelnummer, Name, Studiengang, Semesterzahl)
- Vorlesung = (Vorlesungsnummer, Titel, Semesterwochenstunden)
- Dozent = (Personalnummer, Name, Rang)
- Prüfungen = (Datum, Fach, Note)
- Beziehungen: liest, prüft, hört, nimmt teil

Vielleicht noch Adresse, für Kontaktaufnahme. Vielleicht noch Geschlecht, für personalisierte Anrede. Vielleicht noch . . .

Und jetzt das nochmal als Diagramm.

Entity-Relationship-Modell

Entitäten, Entitätstypen und Attribute
Beziehungen
Erweiterungen

Texte führten auf Objekte, Beziehungen und Attribute. Jetzt Überführen in ein Diagramm.

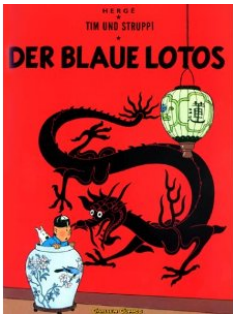
Neue Strukturen dafür:

- *‚Entities‘* (Gegenstände, Entitäten),
- *‚Relationships‘* (Beziehungen, Relationen),
- *‚Attributes‘* (Eigenschaften, Attribute), und
- *‚Roles‘* (Rollen).

Entitäten, Entitätstypen und Attribute

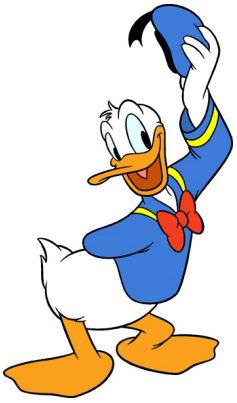
Konkrete Objekte aus der Miniwelt: einzelner Comic, einzelne Studierende, . . .

> ‚*Entität*‘: individuelles, eindeutig identifizierbares Objekt, das durch Eigenschaften charakterisiert werden kann.



Mein ‚Tim&Struppi: Der Blaue Lotos‘ ist eine Entität. Wird beschrieben über: Titel, Serie, Nr., Jahr, Autor, . . .

Student Donald Duck mit Matrikelnummer `01234` und Semesterzahl von 18 im Studiengang ‚Müßiggang‘ ist eine Entität.



Eigenschaften werden Attribute genannt.

> ‚*Attribut*‘: besteht aus einem Namen und einem Datentyp. Der Wertebereich eines Attributs heißt ‚*Domäne*‘. Eine konkrete Ausprägung eines Attributs heißt ‚*Wert*‘.

$dom(A)$: Wertebereich des Attributs A.

Attribute modellieren Eigenschaften von Entitäten und Beziehungen.

‚Titel‘ ist Name eines Attribut eines Comics. Datentyp: `string` (oder `VARCHAR(100)`). Wertebereich: alle Zeichenketten mit 100 Zeichen. Wert: `Der blaue Lotos`.

‚Matrikelnummer‘ ist Name eines Attributs für Studierende. Datentyp: `integer`. Domäne: alle (auf dem Computer darstellbaren) ganzen Zahlen. Wert: `01234` für Donald Duck.

Für Studierende der HS Bremerhaven: $dom(\text{Matrikel}\#) \subseteq [10\,000, 40\,000] \subset \mathbb{N}$.

Wertebereich nutzen, um Korrektheit der Daten zu prüfen: Abfragen.

Attribut kann *‚einwertig'* oder *‚mehrwertig'*/*‚mengenwertig'* sein.

- Studierende können mehrere ‚Telefonnummern' haben.
- Sie haben aber nur eine ‚Matrikelnummer'.

‚Zusammengesetztes' (*‚strukturiertes'*) Attribut besteht aus mehreren einzelnen Attributen.

- ‚Name' meines Freundes/meiner Freundin besteht aus ‚Vorname', ‚Mittelname' und ‚Nachname'.
- Seine/Ihre ‚Adresse' besteht aus ‚Straße', ‚Hausnummer', ‚Postleitzahl' und ‚Ort'.

Mehwertige und zusammengesetzte Attribute müssen für DB vermieden bzw. korrigiert werden.
↪ Normalformen

‚Schlüssel': Eine Menge von (zusammengesetztes) Attribut, deren Wert eine Entität eindeutig beschreibt.

Bspw. *alle* Attribute beschreiben Entität eindeutig. Nicht minimal.

Bei nicht zu vielen Freunden dient ‚Name' als Schlüssel. Sollten es sehr viele werden, dann durchzählen mit ‚FNr': künstlicher Schlüssel.

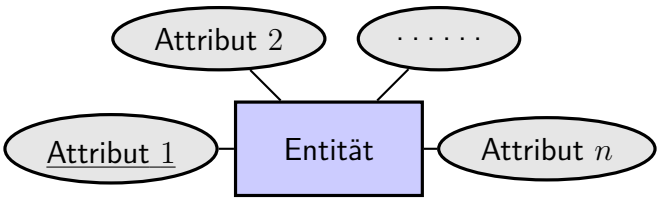
Comic wird identifiziert über ‚Titel', ‚Serie' und ‚Nr.'. Für besondere Ausgaben noch Attribut ‚Jahr'. Künstlicher Schlüssel ‚CNr.' sinnvoll.

Suchen später minimale Schlüssel. Einer davon wird *‚primär Schlüssel'*.

Für DB-Entwurf individuelle Entität unbedeutend. Wichtig ist die zugehörige ‚Schablone'.

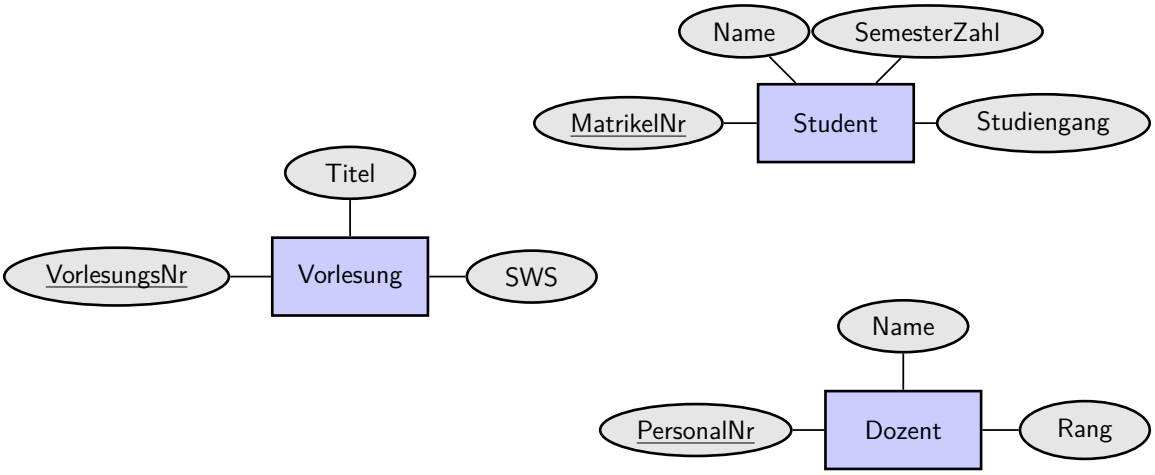
> *‚Entitätstyp'*: Schablone mit eindeutigem Namen. In der Schablone werden alle Attribute zusammengefasst, die die individuelle Entität ausmacht.

Entitätstypen in einem Rechteck dargestellt, zugehörige Attribute in Ellipsen, verbunden mit der Entität.



Namen für Entitätstypen und Attribute sind (grossgeschriebene) Substantive im Singular.

Schlüssel-Attribute werden unterstrichen.



Attribut-Name innerhalb Entitätstypes eindeutig. Gleichnamige Attribute in verschiedenen Entitätsdiagrammen erlaubt.
In Tabellen für RDBMS daraus aufpassen.

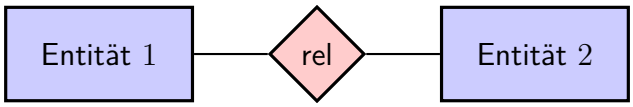
Als nächstes: Studenten *hören* Vorlesungen, Dozenten *lesen* Vorlesungen, . . .

Beziehungen

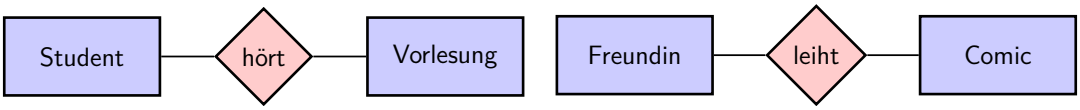
Objekte der Miniwelt stehen in Verbindung miteinander. Genauso Entitäten/Entitätstypen.

‚Beziehungstyp': verknüpft verschiedene Entitätstypen in einer Beziehung.

- Studierende *hören* verschiedene Vorlesungen.
- Freunde *leihen* sich mehrere Comics.



Verb in Raute beschreibt Relation.

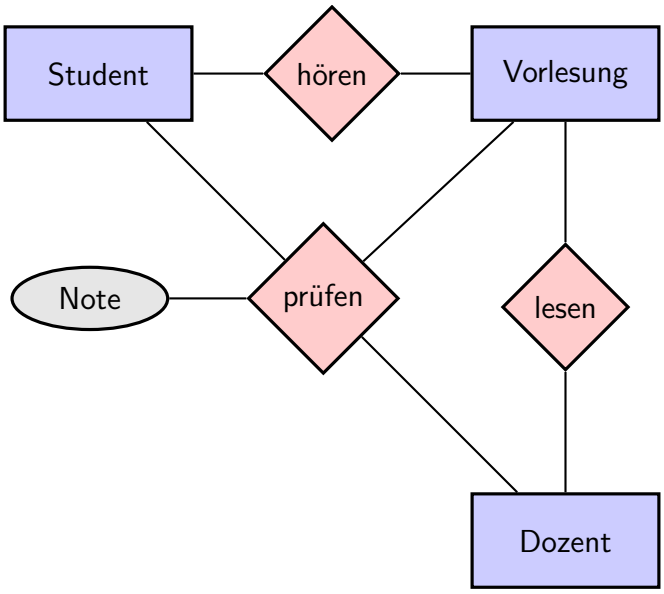


Beziehung sollte gut von links nach rechts zu lesen sein. Meist auch sinnvoll in anderer Richtung.

Wenn nicht, dann Aliasnamen ergänzen.

Einzelner Comic-Titel *gehört zu* einer Comic-Serie. Comic-Serie *besteht aus* einzelnen Comic-Heften.

Alias: besteht_aus



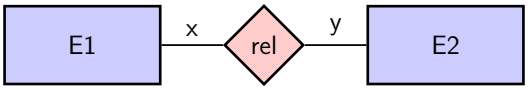
Auch Beziehungstypen können Attribute besitzen.

Aktion: ER-Modell für Comics, Freunde und ausleihen mit Angabe von Ausleihdatum und Rückgabedatum.

- Ein Studierender kann (oder muss) mehrere Vorlesungen hören.
- Eine Vorlesung wird i.a. von mehrerer Studierenden gehört.

- Eine Vorlesung wird von genau einem Dozenten gelesen.
- Ein Dozent kann mehrere Vorlesungen halten.

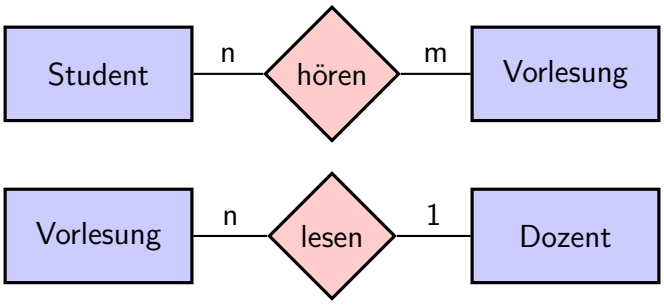
‚Kardinalitäten': geben an, wieviele der einen Entität mit wievielen der anderen Entität in Beziehung stehen. Und umgekehrt.



Von links nach rechts
Eine Entität vom Typ E1 steht in der rel-Beziehung zu y Entitäten des Typs E2.

Von rechts nach links
Eine Entität vom Typ E2 steht in der rel-Beziehung zu x Entitäten des Typs E1.

Vorsicht: Reihenfolge nicht immer eindeutig!

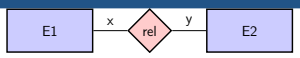


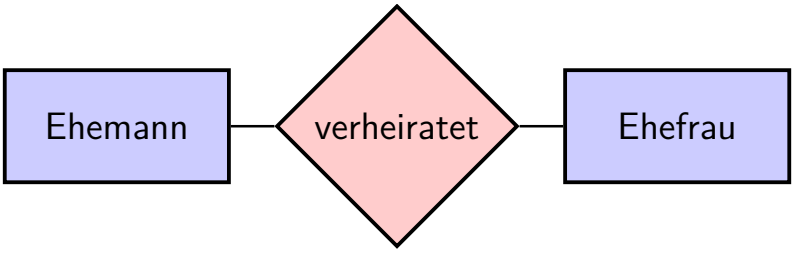
Ein Student hört m Vorlesungen und eine Vorlesung wird von n Studenten besucht.

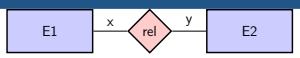
Eine Vorlesung wird von einem Dozenten gelesene und ein Dozent liest insgesamt n Vorlesungen.

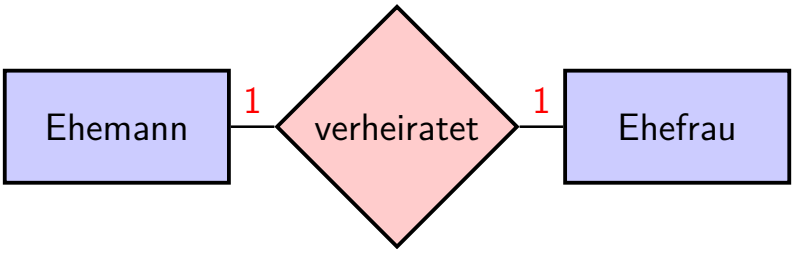
Vier verschiedene Kardinalitäten:

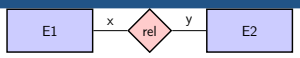
1 eine Beziehung zu genau einer Entität,
C eine Beziehung zu einer oder keiner Entität,
N eine Beziehung zu einer oder mehreren Entitäten,
NC eine Beziehung zu keiner, einer oder mehreren Entitäten.

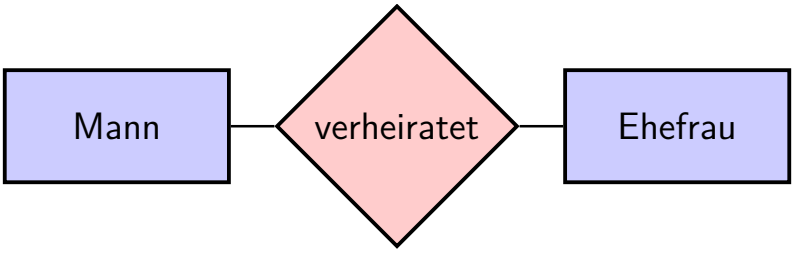


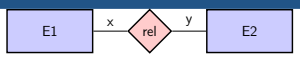


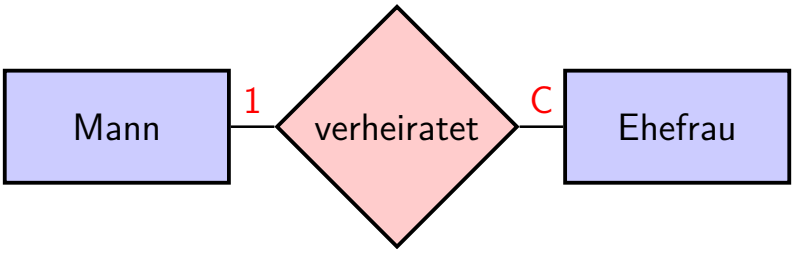

E1
x
rel
y
E2

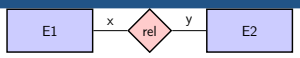

Ehemann
1
verheiratet
1
Ehefrau

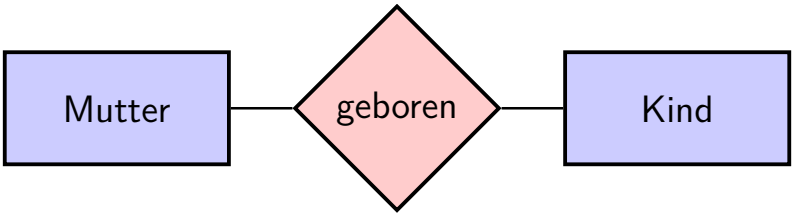

E1
x
rel
y
E2

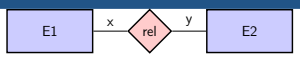

Mann
verheiratet
Ehefrau

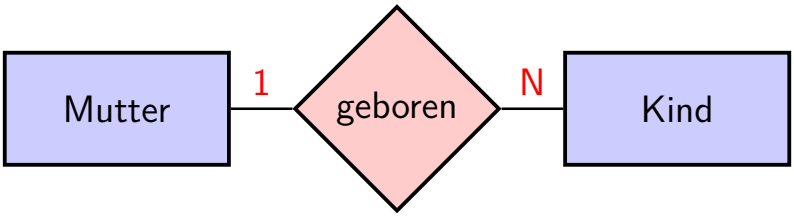

E1
x
rel
y
E2


Mann
1
verheiratet
C
Ehefrau

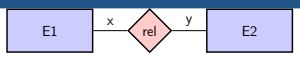

E1
x
rel
y
E2

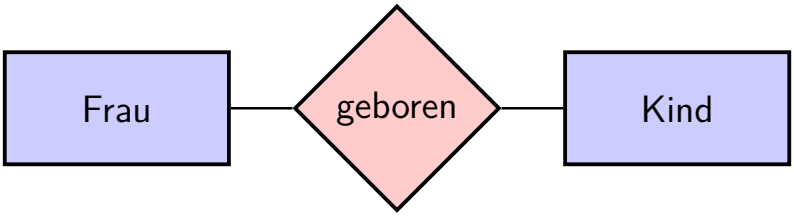

Mutter
geboren
Kind



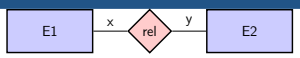


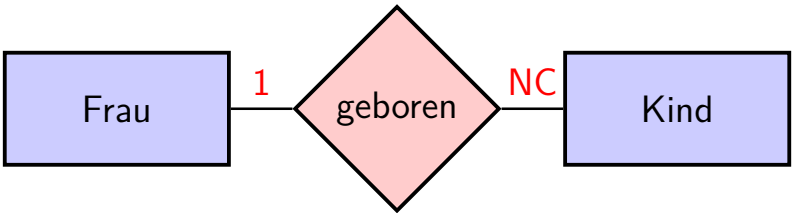
Entity-Relationship-Modell Beziehungen



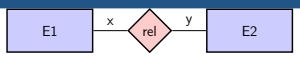


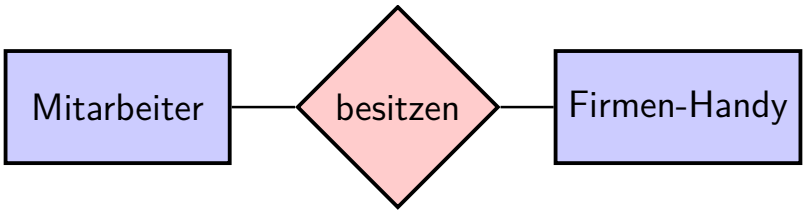
richard rascher-friesenhausen • Datenbanken — WS 23/24 ◁ ▷



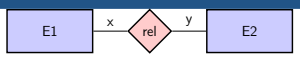


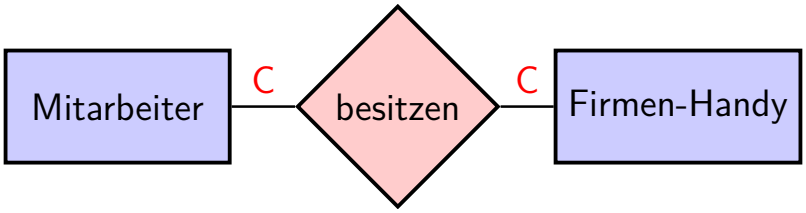
Entity-Relationship-Modell Beziehungen

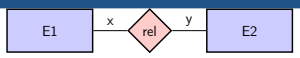


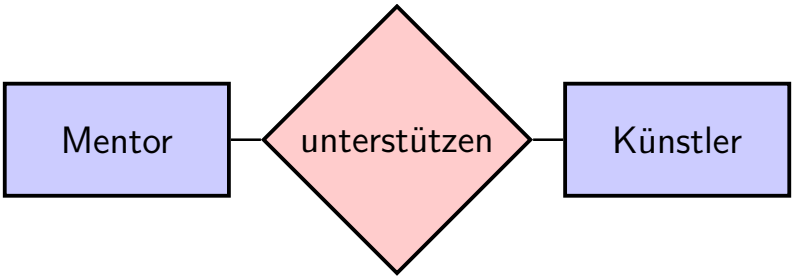


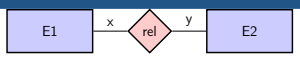
richard rascher-friesenhausen • Datenbanken — WS 23/24 ◁ ▷

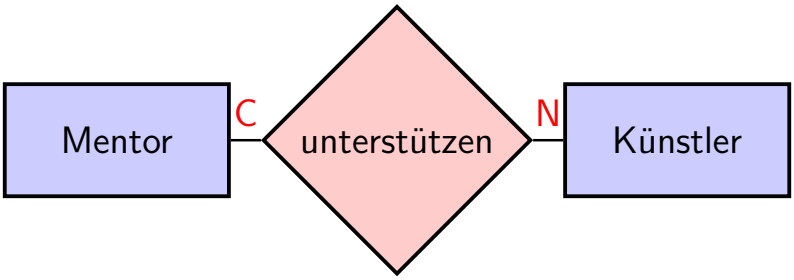

E1
x
rel
y
E2

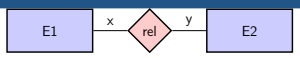

Mitarbeiter
C
besitzen
C
Firmen-Handy

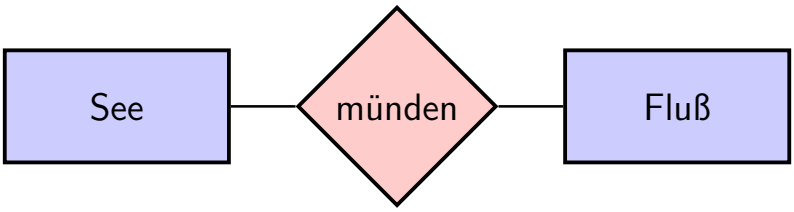

E1
x
rel
y
E2

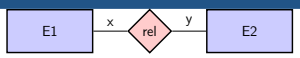

Mentor
unterstützen
Künstler

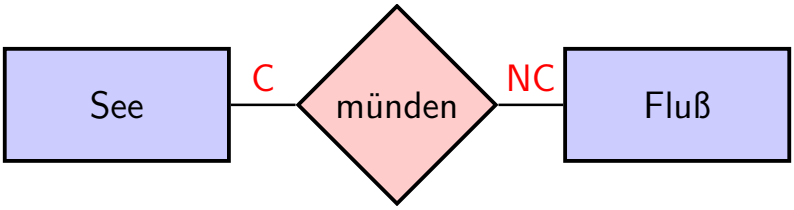

E1
x
rel
y
E2


Mentor
C
unterstützen
N
Künstler

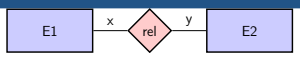

E1
x
rel
y
E2

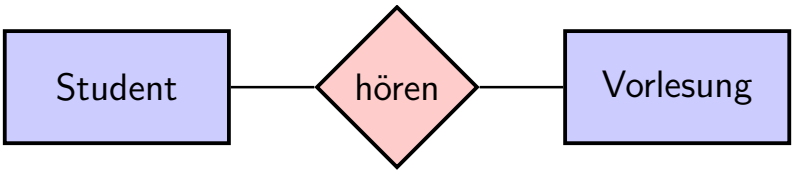

See
münden
Fluß



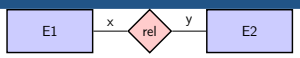


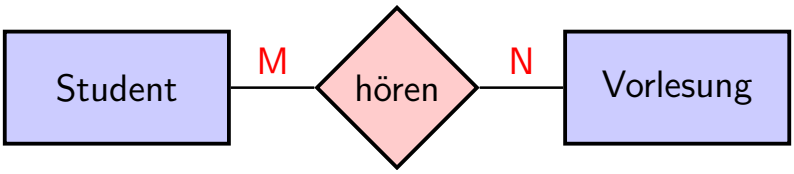
Entity-Relationship-Modell Beziehungen

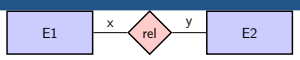


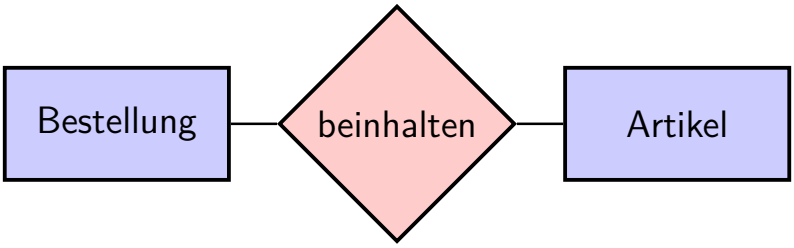


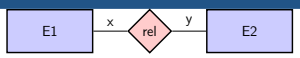
richard rascher-friesenhausen • Datenbanken — WS 23/24 ◁ ▷

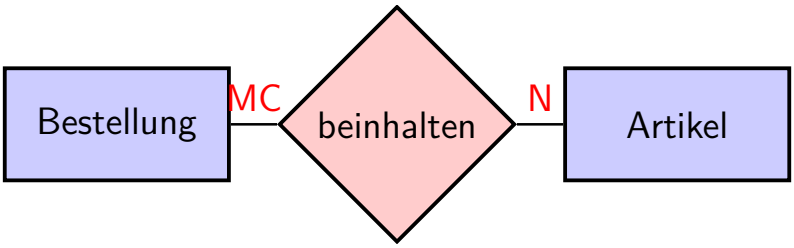

E1
x
rel
y
E2

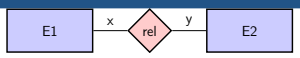

Student
M
hören
N
Vorlesung

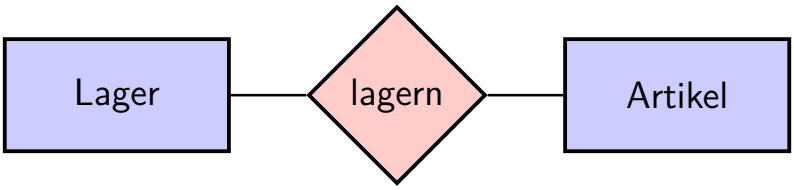

E1
x
rel
y
E2

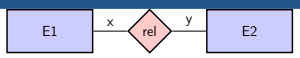

Bestellung
beinhalten
Artikel

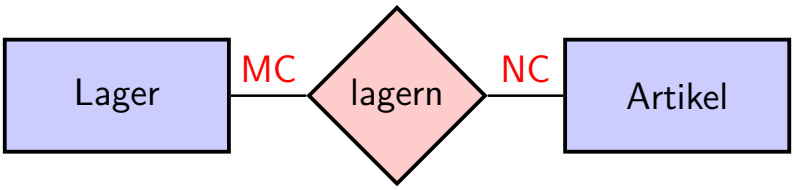

E1
x
rel
y
E2

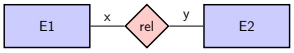

Bestellung
MC
beinhalten
N
Artikel


E1
x
rel
y
E2


Lager
lagern
Artikel







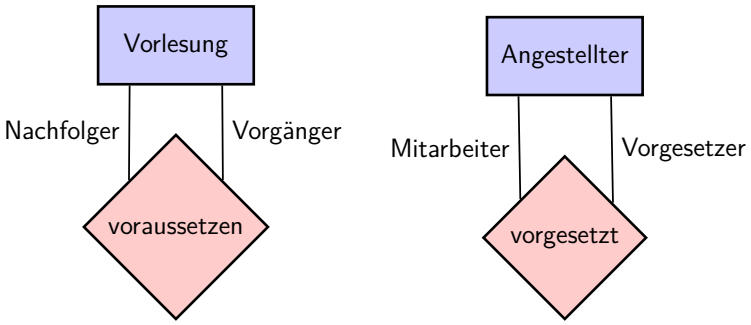
E1	E2	rel	x	y
Ehemann	Ehefrau	verheiratet	1	1
Mann	Ehefrau	verheiratet	1	C
Mutter	Kind	geboren	1	N
Frau	Kind	geboren	1	NC
Mitarbeiter	Firmen-Handy	besitzen	C	C
Mentor	Künstler	unterstützen	C	N
Fluß	See	münden	NC	C
Student	Vorlesung	teilnehmen	M	N
Bestellung	Artikel	beinhalten	NC	M
Artikel	Lager	lagern	MC	NC

Rollen

Es kann auch Beziehungen von Entitäten eines Entitätstyps untereinander geben: *‚rekursive Beziehung'*.

Etwa:

- Eine Vorlesung hat als Voraussetzung den Besuch anderer Vorlesungen.
- Der Vorgesetzte eines Angestellten ist ein Angestellter.
- ...



Wer was ist, wird als *‚Rolle'* notiert.

Erweiterungen

Es existieren Erweiterungen, die ER-Diagramme ausdruckstärker machen.

- Abhängige Entitätstypen
- Generalisierungen und Spezialsierungen
- Optionale Attribute
- Strukturierte Attribute
- Mehrwertige Attribute
- Virtuelle Attribute
- ...

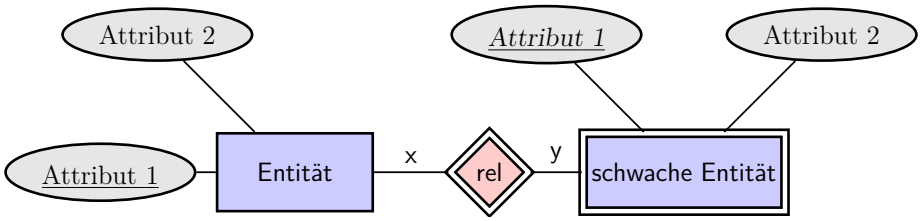
Schaun wir uns kurz an.

Schwache Entitätstypen

‚Schwacher Entitätstyp': existiert nur in Abhängigkeit von einem anderen Entitätstyp.

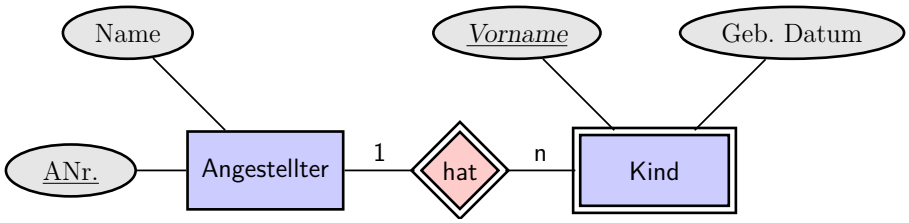
Angestellte der Hochschule können Eltern sein. Wollen auch deren Kinder speichern. Entität ‚Kind' existiert abhängig von Entität ‚Angestellter': *‚schwache Entität'*.

Schwache Entität wird über Beziehungstyp an stärker Entität gebunden. Kennzeichnung über doppelte Linien.



Schwache Entität ohne eigenen Schlüssel. Verwendet Schlüssel der stärkeren Entität (und eigenen Teilschlüssel).

Konkret für unser Kinder-Beispiel:



Schlüssel für ‚Kind': ‚ANr.' & ‚Vorname'.

Für Datenbank: wenn Angestellter gelöscht wird, können auch Kinder automatisch gelöscht werden.

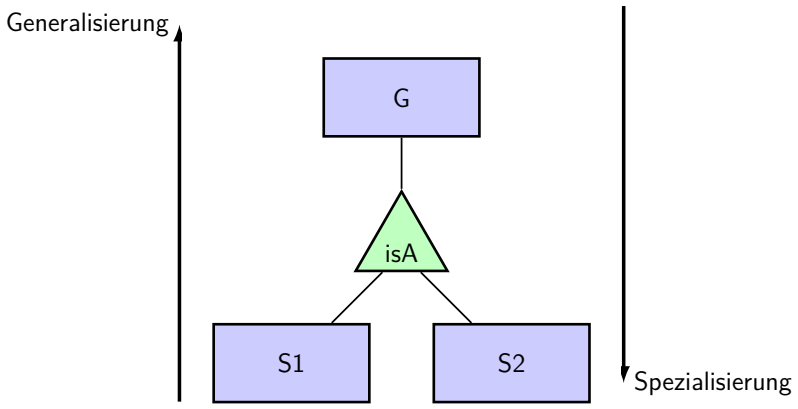
Aufgabe: ER-Diagramm für Gebäude und Räume.

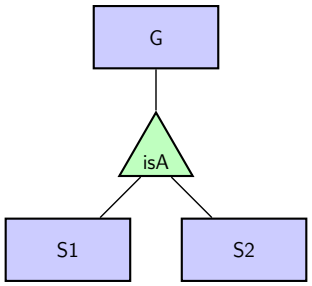
Generalisierung und Spezialisierung

‚*Generalisierung*‘: fasst Gemeinsamkeiten von Entitätstypen in neuem Entitätstyp zusammen.

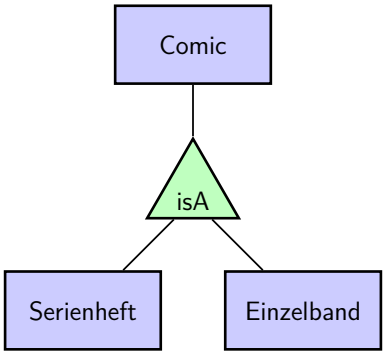
‚*Spezialisierung*‘: erweitert vorhandenen Entitätstyp um weitere Attribute.

Stellen dies über spezielle Beziehung ‚isA‘ dar.





G enthält Attribute, die beide Entitätstypen S1 und S2 gemeinsam haben. Bei Spezialisierung S1 und S2 kommen möglicherweise verschiedene Attribute hinzu.



Man unterscheidet

- ‚*totale*‘ und ‚*partielle*‘,
- ‚*disjunkte*‘ und ‚*nicht disjunkte*‘ Spezialisierungen.

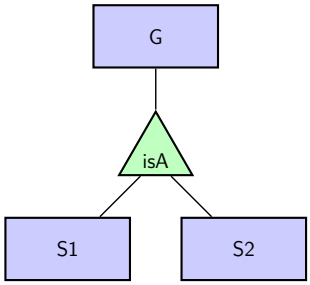
Heißen auch ‚*vollständig*‘, ‚*unvollständig*‘ bzw. ‚*nicht überlappend*‘, ‚*überlappend*‘. Sind unabhängig voneinander.

partiell und disjunkt

G: Hochschulangehörige
S1: Studierende
S2: Dozenten

partiell und nicht disjunkt

G: Studierende
S1: ASTA-Mitglied
S2: Hiwis

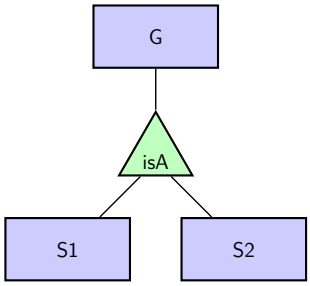


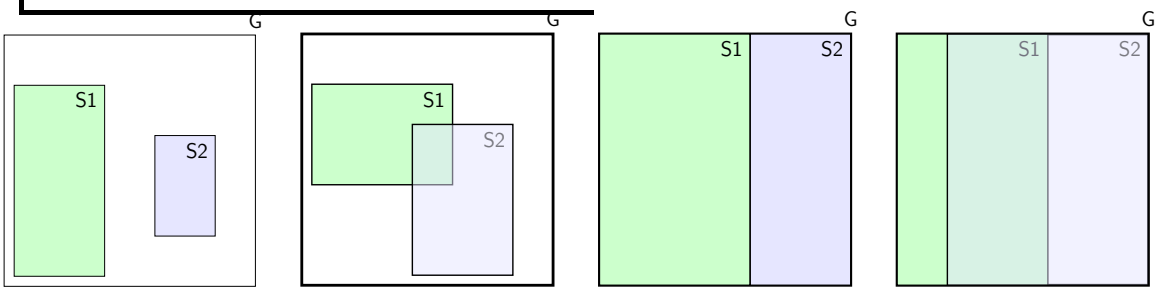
total und disjunkt

G: Menschen
S1: Frauen
S2: Männer

total und nicht disjunkt

G: Bauteile
S1: Fremdteile
S2: Eigenteile

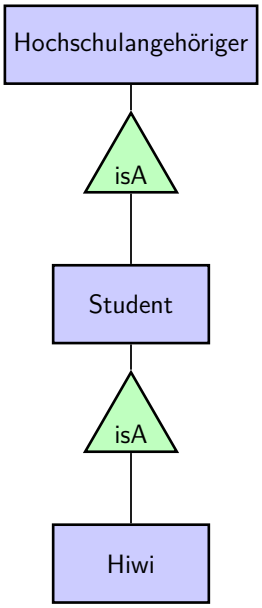




pd pü td tü

isA Beziehung kann über mehrere Ebenen verlaufen:

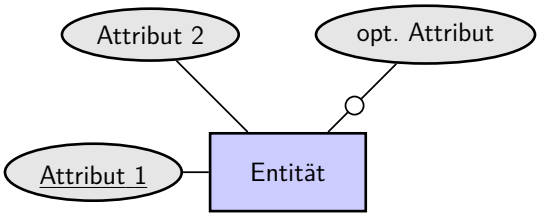
Hiwi isA Student isA Hochschulangehöriger



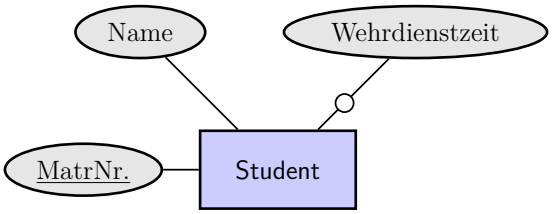
Optionale Attribute

Manch Attribut existiert für die eine Entität, für die andere nicht: *‚optionales Attribut'*.

Wird gekennzeichnet durch kleinen Kreis an der Verbindungslinie.



Bsp.: Wehrdienstzeit bei Studierenden.



Optionale Attribute dürfen nicht zu einem Schlüssel gehören!

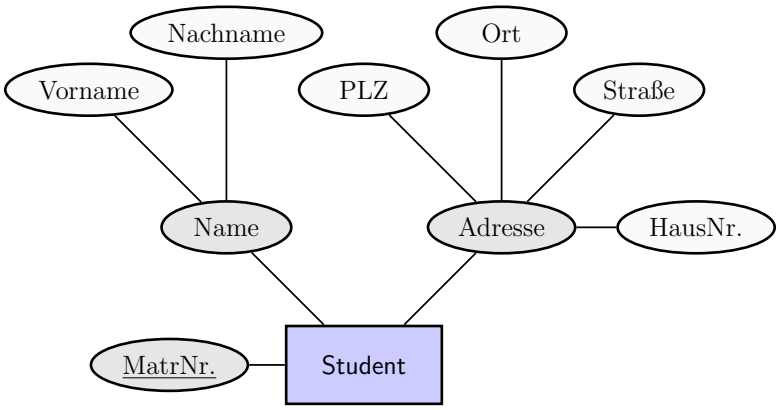
Strukturierte Attribute

Hatten wir schon: manch Attribut ist zusammengesetzt aus verschiedenartigen Teilattributen (nicht atomar).

Bsp.: Addresse = PLZ + Ort + Straße + Hausnummer

Bsp.: Name = Vorname + Mittelname + Nachname

Wird gekennzeichnet durch Anhängen der Teilattribute an das Attribut.



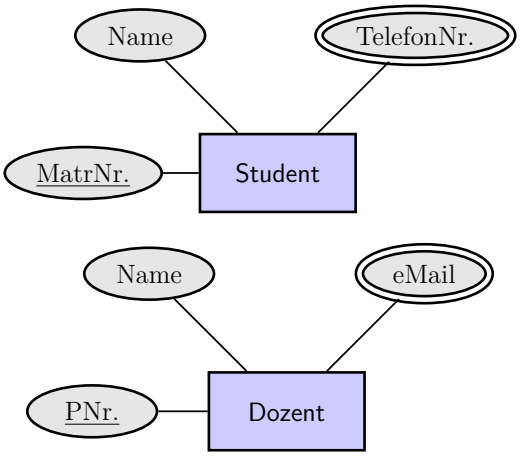
Mehrwertige Attribute

Hatten wir schon: manch Attribut kann mehrere gleichartige Werte annehmen.

Bsp.: Dozent hat mehrere eMail Adressen.

Bsp.: Student hat mehrere Telefonnummern.

Wird gekennzeichnet durch doppelte Umrahmung des Attributs.



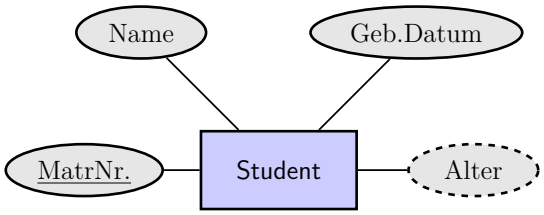
Virtuelle Attribute

Attribut *‚virtuell‘* (*‚abgeleitet‘*, *‚berechnet‘*), wenn aus anderen Attributwerten berechenbar. Angabe der Berechnungsformel notwendig.

Bsp.: Alter kann aus Geburtsdatum berechnet werden:

$$\text{Alter} := \text{Today} - \text{Geburtsdatum}$$

Wird gekennzeichnet durch gestrichelte Umrahmung des Attributs.



Aufgabe: ER-Diagramm Bestellung, Artikel, welches Gesamtpreis enthält.

Relationales Datenbankschema, Tabellen

Tabellen
Relationen
Übersetzung der ER-Diagramme
Vereinfachen der Relationenschemata

Übertragen Entity-Relationship-Diagramme in *‚Tabellen‘*/*‚Relationen‘*. Nächster Schritt zur Datenbank auf dem Rechner.

Formaler Prozess mit mathematischen Hintergrund.