Datenbanken 1
Das Relationale Modell

Nikolaus Augsten

nikolaus.augsten@sbg.ac.at

FB Computerwissenschaften
Universität Salzburg

Sommersemester 2014

Inhalt

Literatur und Quellen

Lektüre zum Thema "Relationales Modell":

- Kapitel 3 (außer 3.5, 3.6) aus Kemper und Eickler: Datenbanksysteme: Eine Einführung. 8. Auflage, Oldenbourg Verlag, 2011.

Literaturquellen

- Elmasri and Navathe: Fundamentals of Database Systems. Fourth Edition, Pearson Addison Wesley, 2004.
- Silberschatz, Korth, and Sudarashan: Database System Concepts, McGraw Hill, 2006.

Danksagung Die Vorlage zu diesen Folien wurde entwickelt von:

- Michael Böhlen, Universität Zürich, Schweiz
- Johann Gamper, Freie Universität Bozen, Italien

Inhalt

Das Relationale Modell/1

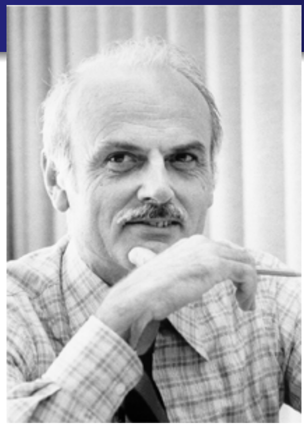
- Relationale Modell:
  - logisches Datenmodell
  - basiert auf Relationen
- Relation: mathematisches Konzept, das auf Mengen basiert.
- Die Stärke des relationalen Modells ist die formale Grundlage durch Relationen (und Mengen).
- In der Praxis wird der SQL Standard verwendet. Der SQL Standard unterscheidet sich vom formalen Modell in einigen Punkten (wir gehen später auf diese Unterschiede ein).

Das Relationale Modell/2

- Das relationale Modell wurde von E. Codd von IBM Research in folgendem Artikel eingeführt:

  A Relational Model for Large Shared Data Banks, Communications of the ACM, June 1970
- Dieser Artikel hat das Feld der Datenbanksysteme revolutioniert.
- Codd erhielt hierfür den ACM Turing Award.

Das Relationale Modell/3



- Edgar Codd, a mathematician and IBM Fellow, is best known for creating the relational model for representing data that led to today's 12 billion database industry.
- Codd's basic idea was that relationships between data items should be based on the item's values, and not on separately specified linking or nesting.
- The idea of relying only on value-based relationships was quite a radical concept at that time, and many people were skeptical. They didn't believe that machine-made relational queries would be able to perform as well as hand-tuned programs written by expert human navigators.

`http://www-03.ibm.com/ibm/history/exhibits/builders/builders_codd.html`

Schema

- $sch(R) = [A_1, A_2, \ldots, A_n]$ ist das Schema der Relation.
- Eckige Klammern [...] werden für eine Liste von Werten verwendet; eine Liste ist geordnet.
- R ist der Name der Relation.
- $A_1, A_2, \ldots, A_n$ sind die Attribute.
- Kurzschreibweise: Für die Definition einer Relation R mit Schema $sch(R) = [A_1, A_2, \ldots, A_n]$ schreiben wir kurz:

$$R[A_1, A_2, \ldots, A_n]$$

- Beispiel: Für die Relation $Kunden[KundenName, KundenStrasse, KundenOrt]$ gilt $sch(Kunden) = [KundenName, KundenStrasse, KundenOrt]$

Die Relation Kunden

- Schema: $sch(Kunden) = [KundenName, KundenStrasse, KundenOrt]$

Kunden

KundenName	KundenStrasse	KundenOrt
Meier	Zeltweg	Brugg
Steger	Ringstr	Aarau
Marti	Seeweg	Brugg
Kurz	Marktplatz	Luzern
Egger	Weststr	Brugg
Staub	Bahnhofstr	Brugg
Gamper	Bahnhofstr	Chur
Ludwig	Baugasse	Brugg
Wolf	Bahnhofstr	Brugg
Koster	Magnolienweg	Brugg
Kunz	Fliedergasse	Brugg
Pauli	Murtenstr	Biel

Domäne

- Eine Domäne ist eine Menge von atomaren Werten.
  - Beispiel: Alter einer Person ist eine positive Ganzzahl.
- Zu jeder Domäne gehört ein Datentyp (oder Format):
  - Telefonnummer hat Format: 0dd ddd dd dd, wobei d eine Ziffer ist.
  - Für ein Datum existieren verschiedene Formate: z.B. yyyy-mm-dd oder dd.mm.yyyy
- Der reservierte Wert `null` gehört zu jeder Domäne:
  - wird für fehlende Werte verwendet
  - Nullwerte machen die Definition von Operationen komplexer

Attribute

- Attributwert: Attribut nimmt für jedes Tupel einen Wert an
  - mögliche Werte durch Domäne bestimmt
  - $dom(A)$ ist die Domäne von Attribut A
- Atomar: Attributwerte müssen atomar sein
  - Der Wert eines Attributs kann eine Kontonummer sein, aber nicht eine Menge von Kontennummern.
- Attributname: spezifizieren Rolle der entsprechenden Domäne in Relation:
  - Name ist eindeutig innerhalb einer Relation
  - wird verwendet, um die Werte dieses Attributs zu interpretieren
- Beispiel: Die Domäne *Datum* wird für die Attribute *Rechnungsdatum* und *Zahlungstermin* mit unterschiedlichen Bedeutungen verwendet.
  - $dom(Rechnungsdatum) = Datum$
  - $dom(Zahlungstermin) = Datum$

Tupel

- Ein Tupel ist eine geordnete Menge (d.h. eine Liste) von Werten
- Eckige Klammern [...] werden verwendet um Tupel darzustellen
- Jeder Wert eines Tupels muss aus der entsprechenden Domäne stammen
- Beispiel: Tupel der Relation *Kunden*
  - Schema: $sch(Kunden) = [KundenName, KundenStrasse, KundenOrt]$
  - Tupel: $[Meier, Zeltweg, Brugg]$

Instanz (Ausprägung)

- Der Name einer Relation R wird auch als Bezeichner für die Instanz einer Relation verwendet
- Instanz einer Relation R mit Schema $sch(R) = [A_1, A_2, \ldots, A_n]$ ist eine Untermenge des Kreuzprodukts der Domänen der Attribute:

$$R \subseteq D_1 \times D_2 \times \ldots \times D_n$$

  R ist also eine Menge von Tupeln $[v_1, v_2, \ldots, v_n]$, sodass jedes $v_i \in D_i$
- Geschweifte Klammern $\{\ldots\}$ werden für Mengen verwendet
- Beispiel:

$$\begin{aligned}
D_1 &= dom(KuName) = \{Ludwig, Koster, Marti, Wolf, \ldots\} \\
D_2 &= dom(KuStrasse) = \{Bahnhofstr, Baugasse, Seeweg, \ldots\} \\
D_3 &= dom(KuOrt) = \{Brugg, Luzern, Chur, \ldots\} \\
R &= \{\ [Ludwig, Bahnhofstr, Brugg], [Koster, Baugasse, Brugg], \\
&\quad\ [Marti, Seeweg, Brugg], [Wolf, Weststr, Brugg]\ \} \\
&\subseteq D_1 \times D_2 \times D_3
\end{aligned}$$

Beispiel einer Relation

Relation Attribute

Tupel

Konten

KoNr	FName	Guthaben
A-101	Chur	500
A-215	Brugg	700
A-102	Brugg	400
A-305	Brugg	350
A-201	Aarau	900
A-222	Brugg	700
A-217	Aarau	750

Eigenschaften von Relationen

- Relationen sind ungeordnet, d.h., die Ordnung der Tupel ist nicht relevant.
- Die Attribute eines Schemas $sch(R) = [A_1, \ldots, A_n]$ und die Werte in einem Tuple $t = [v_1, \ldots, v_n]$ sind geordnet.

Konten

KoNr	FName	Guthaben
A-101	Chur	500
A-215	Brugg	700
A-102	Brugg	400
A-305	Brugg	350
A-201	Aarau	900
A-222	Brugg	700
A-217	Aarau	750

=

Konten

KoNr	FName	Guthaben
A-305	Brugg	350
A-201	Aarau	900
A-222	Brugg	700
A-102	Brugg	400
A-217	Aarau	750
A-101	Chur	500
A-215	Brugg	700

Integrierte Übung 3.1

1. Ist $R = \{[Tom, 27, ZH], [Bob, 33, Rome, IT]\}$ eine Relation?
2. Was ist der Unterschied zwischen einer Menge und einer Relation? Geben Sie ein Beispiel, das den Unterschied illustriert.

Datenbank

- Eine Datenbank ist eine Menge von Relationen.
- Beispiel: Die Informationen eines Unternehmens werden in mehrere Teile aufgespaltet:
  - *Konten*: speichert Informationen über Konten
  - *Kunde*: speichert Informationen über Kunden
  - *Kontoinhaber*: speichert welche Kunden welche Konten besitzen
- Warum nicht alle Informationen in eine Relation speichern?
  - Beispiel: $sch(Bank) = [KoNr, Guthaben, KuName, \ldots]$
  - Redundanz: Wiederholung von Informationen, z.B. zwei Kunden mit demselben Konto
  - Nullwerte: z.B. für einen Kunden ohne Konto

Datenbank mit Relationen Konten und Kontoinhaber

Konten

KoNr	FName	Guthaben
A-101	Chur	500
A-215	Brugg	700
A-102	Brugg	400
A-305	Brugg	350
A-201	Aarau	900
A-222	Brugg	700
A-217	Aarau	750

Kontoinhaber

KName	KoNr
Staub	A-102
Gamper	A-101
Gamper	A-201
Ludwig	A-217
Wolf	A-222
Koster	A-215
Kunz	A-305
Mair	A-101

Integrierte Übung 3.2

1. Illustrieren Sie die folgenden Relationen graphisch:
   $sch(R) = [X, Y]; R = \{[1, a], [2, b], [3, c]\}$
   $sch(S) = [A, B, C]; S = \{[1, 2, 3]\}$

2. Bestimmen Sie die folgenden Objekte:
   - Das 2. Attribut der Relation R?
   - Das 3. Tupel der Relation R?
   - Das Tuple in der Relation R mit dem kleinsten Wert von Attribut X?

Zusammenfassung des relationalen Modells

- Eine **Domäne** D ist eine Menge von atomaren Werten.
  - Telefonnummern, Namen, Noten, Geburtstage, Institute
  - jede Domäne beinhaltet den reservierten Wert `null`
- Zu jeder Domäne wird ein **Datentyp** oder Format spezifiziert.
  - 5-stellige Zahlen, yyyy-mm-dd, Zeichenketten
- Ein **Attribut** A_i beschreibt die Rolle einer Domäne innerhalb eines Schemas.
  - TelephonNr, Alter, Institutsname
- Ein **Schema** $sch(R) = [A_1, ..., A_n]$ besteht aus einer Liste von Attributen.
  - $sch(Angestellte) = [Name, Institut, Lohn]$, $sch(Institute) = [InstName, Leiter, Addresse]$
- Ein **Tupel** t ist eine Liste von Werten $t = [v_1, ..., v_n]$ mit $v_i \in dom(A_i)$.
  - $t = [Tom, SE, 23K]$
- Eine **Relation** $R \subseteq D_1 \times ... \times D_n$ mit dem Schema $sch(R) = [A_1, ..., A_n]$ ist eine Menge von n-stelligen Tupeln.
  - $R = \{[Tom, SE, 23K], [Lene, DB, 33K]\} \subseteq NAMEN \times INSTITUTE \times INTEGER$
- Eine **Datenbank** DB ist eine Menge von Relationen.
  - $DB = \{R, S\}$
  - $R = \{[Tom, SE, 23K], [Lene, DB, 33K]\}$
  - $S = \{[SE, Tom, Boston], [DB, Lena, Tucson]\}$

Integrierte Übung 3.3

1. Welche Art von Objekt ist $X = \{\{[3]\}\}$ im relationalen Modell?

2. Sind DB1 und DB2 identische Datenbanken?
   $DB1 = \{\{[1,5],[2,3]\},\{[4,4]\}\}$
   $DB2 = \{\{[4,4]\},\{[2,3],[1,5]\}\}$

Integritätsbedingungen

- Integritätsbedingungen (constraints) sind Einschränkungen auf den Daten, die alle Instanzen der Datenbank erfüllen müssen.
- Klassen von Integritätsbedingungen im relationalen Modell :
  - Schlüssel
  - Domänenintegrität
  - Referentielle Integrität
- Integritätsbedingungen garantieren eine gute Datenqualität.

Schlüssel/1

- $K \subseteq R$ ist eine Teilmenge der Attribute von R
- K ist ein Superschlüssel von R falls die Werte von K ausreichen um ein Tupel jeder möglichen Relation R eindeutig zu identifizieren.
  - Mit "jeder möglichen" meinen wir eine Relation, die in der Miniwelt, die wir modellieren, existieren könnte.
  - Beispiel: $\{KuName, KuStrasse\}$ und $\{KuName\}$ sind Superschlüssel von *Kunde*, falls keine zwei Kunden den gleichen Namen haben können.

KuName	KuStrasse
N. Jeff	Binzmühlestr
N. Jeff	Hochstr

KuName ist *kein* Schlüssel

KuName	KuStrasse
N. Jeff	Binzmühlestr
T. Hurd	Hochstr

KuName ist ein Schlüssel

Schlüssel/2

- K ist ein Kandidatschlüssel falls K minimal ist
  Beispiel:
  - $\{KuName\}$ ist ein Kandidatschlüssel für *Kunde* weil diese Menge ein Superschlüssel ist und keine Untermenge ein Superschlüssel ist.
  - $\{KuName, KuStrasse\}$ ist kein Kandidatschlüssel weil eine Untermenge, nämlich $\{KuName\}$, ein Superschlüssel ist.
- Primärschlüssel: ein Kandidatschlüssel der verwendet wird um Tupel in einer Relation zu identifizieren.
  - Als Primärschlüssel sollte ein Attribut ausgewählt werden, dessen Wert sich nie ändert (oder zumindest sehr selten).
  - Beispiel: *email* ist eindeutig und ändert sich selten

Domänenintegrität

- Die Domänenintegrität garantiert, dass alle Attributwerte aus der entsprechenden Domäne stammen.
- Nullwerte: sind standardmäßig erlaubt da Teil der Domäne
- Primärschlüssel dürfen nicht `null` sein
  - falls der Primärschlüssel aus mehreren Attributen besteht darf keines dieser Attribute `null` sein
  - andere Attribute der Relation, selbst wenn sie nicht zum Primärschlüssel gehören, können ebenfalls Nullwerte verbieten

ID	Name	KuStrasse
1	N. Jeff	Binzmühlestr
`null`	T. Hurd	Hochstr

ID kann nicht Primärschlüssel sein

ID	Name	KuStrasse
1	N. Jeff	Binzmühlestr
2	T. Hurd	Hochstr

ID kann Primärschlüssel sein

Referentielle Integrität

- Fremdschlüssel: Attribute im Schema einer Relation, die Primärschlüssel einer anderen Relation sind.
  - Beispiel: *KuName* und *KoNr* der Relation *Kontoinhaber* sind Fremdschlüssel von *Kunde* bzw. *Konten*.
- Rekursion: Nicht-Primärschlüssel Attribute können auch Fremdschlüssel zum Primärschlüssel in derselben Relation sein.
- Erlaubte Werte für Fremdschlüssel:
  - Werte, die als Primärschlüssel in der referenzierten Relation vorkommen
  - `null` Werte (alle oder kein Attribut des Fremdschlüssels)
- Graphischen Darstellung eines Schemas: gerichteter Pfeil vom Fremdschlüsselattribut zum Primärschlüsselattribut.

ID	KuName	KuStrasse
1	N. Jeff	2
2	T. Hurd	4

StrassenNr	Strasse
2	Binzmühlestr
3	Hochstr

KuStrNr kann kein Fremdschlüssel sein weil StrassenNr 4 nicht existiert.

Integrierte Übung 3.4

- Bestimmen Sie die Schlüssel der Relation *R*:

R

X	Y	Z
1	2	3
1	4	5
2	2	2

Integrierte Übung 3.5

- Bestimmen Sie mögliche Superschlüssel, Kandidatschlüssel, Primärschlüssel und Fremdschlüssel für die Relationen *R* und *S*:

R

A	B	C
a	d	e
b	d	c
c	e	e

S

D	E
d	a
e	a
a	a

mögliche Superschlüssel:

mögliche Kandidatschlüssel:

mögliche Primärschlüssel:

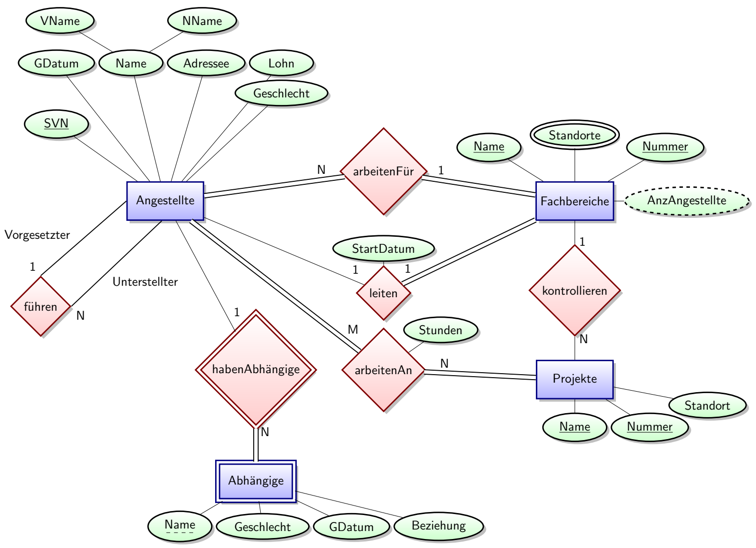
mögliche Fremdschlüssel:

Inhalt

Algorithmus ER-Schema → Relationales Modell

- Algorithmus um anhand eines konzeptionallen ER Schemas (fast) automatisch ein relationales Schema zu erstellen.
  - Schritt 1: Abbildung von unabhängigen Entitätstypen
  - Schritt 2: Abbildung von existenzanhängigen Entitätstypen
  - Schritt 3: Abbildung von 1:1 Beziehungentypen
  - Schritt 4: Abbildung von 1:N Beziehungstypen
  - Schritt 5: Abbildung von M:N Beziehungstypen
  - Schritt 6: Abbildung von mehrwertigen Attributen
  - Schritt 7: Abbildung von n-wertigen Beziehungstypen
  - Schritt 8: Abbildung von Spezialisierung/Generalisierung

Beispiel: ER Schema der NAWI Datenbank



Schritt 1: Abbildung unabhängiger Entitätstypen

(a) Entitätstyp: Für jeden unabhängigen Entitätstypen E erstellen wir eine Relation R.

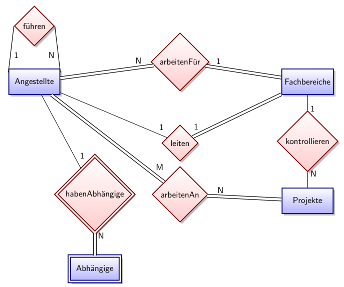
(b) Attribute: Die Attribute von R sind
- alle einfachen Attributen von E
- alle einfache Komponenten von zusammengesetzten Attributen

(c) Primärschlüssel: Ein Schlüsselattribut von E wird als Primärschlüssel für R ausgewählt.

→ Falls der ausgewählte Schlüssel von E zusammengesetzt ist, besteht der Primärschlüssel aus allen einfachen Komponenten.

Beispiel: Abbildung unabhängiger Entitätstypen

- **Beispiel:** Wir erstellen Relationen Angestellte, Fachbereiche, Projekte.
  - SVN, FNummer, und PNummer sind die Primärschlüssel



Angestellte[VName, NName, <u>SVN</u>, GDatum, Adresse, Geschlecht, Lohn]
Fachbereiche[FName, <u>FNummer</u>]
Projekte[PName, <u>PNummer</u>, PStandort]

Schritt 2: Abbildung existenzabhängiger Entitätstypen

(a) Existenzabhängiger Entitätstyp: Für jeden existenzabhängigen Entitätstypen W mit übergeordnetem Entitätstypen E erstellen wir eine Relation R.

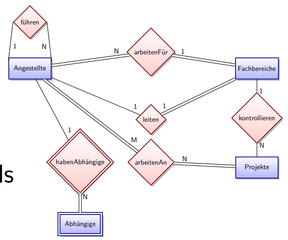
(b) Attribute von R sind alle einfachen Attribute bzw. einfachen Komponenten zusammengesetzter Attribute von W.

(c) Fremdschlüssel: Der Primärschlüssel der Relation des übergeordneten Entitätstypen E wird als Fremdschlüssel zu R hinzugefügt.

(d) Primärschlüssel von R besteht aus der *Kombination* der
- Primärschlüssel der übergeordneten Entitätstypen
- des partiellen Schlüssels des existenzabhängigen Entitätstypen

Beispiel: Abbildung existenzabhängiger Entitätstypen

- **Beispiel:** Der existenzabhängigen Entitätstypen Abhängige wird auf Relation Abhängige abbildet.
  - Primärschlüssel SVN von Angestellte wird als Fremdschlüssel zu Relation Abhängige hinzugefügt (umbenannt auf AngSVN).
  - Der Primärschlüssel von Abhängige ist die Kombination { AngSVN, AbhName }, weil AbhName ein partieller Schlüssel von Abhängige ist.



Angestellte[VName, NName, <u>SVN</u>, GDatum, Adresse, Geschlecht, Lohn]
Fachbereiche[FName, <u>FNummer</u>]
Projekte[PName, <u>PNummer</u>, PStandort]
Abhängige[<u>AngSVN</u>, <u>AbhName</u>, Geschlecht, GDatum, Beziehung]

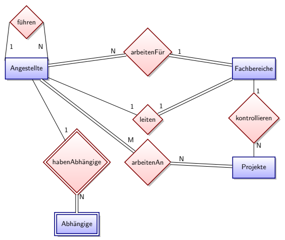
Schritt 3: Abbildung von 1:1 Beziehungstypen

(a) Involvierte Entitätstypen: Für jeden 1:1 Beziehungstypen im ER Schema identifizieren wir die Relationen S und T der involvierten Entitätstypen, z.B. Angestellte leiten Fachbereiche

(b) Es existieren drei mögliche Ansätze:
1. **Zusammengefasste Relationen:** Beide beteiligten Entitätstypen sowie der 1:1 Beziehungstyp werden in einen einzigen Entitätstypen zusammengelegt. Kann sinnvoll sein, wenn beide Beziehungen total sind.
2. **Fremdschlüssel:** Eine der beteiligten Relationen wird ausgewählt, z.B. S, und der Primärschlüssel von T wird als Fremdschlüssel zu S hinzugefügt. Verhindert Null-Werte, wenn S eine totale Beziehungen eingeht.
3. **Neue Beziehungsrelation:** Neue Relation R mit den Primärschlüsseln von S und T als Fremdschlüssel. Einer der beiden Fremdschlüssel von R ist Primärschlüssel. Verhindert Null-Werte, wenn keine der Beziehungen total ist.

Beispiel: Abbildung von 1:1 Beziehungstypen

- **Beispiel**: Der 1:1 Beziehungstyp leiten wird mithilfe eines Fremdschlüssels abgebildet. Fachbereiche übernimmt die Rolle von S, weil die Teilnahme in der Beziehung total ist.



Angestellte[VName, NName, <u>SVN</u>, GDatum, Adresse, Geschlecht, Lohn]
Fachbereiche[FName, <u>FNummer</u>, LeiterSVN, StartDatum]
Projekte[PName, <u>PNummer</u>, PStandort]
Abhängige[<u>AngSVN</u>, <u>AbhName</u>, Geschlecht, GDatum, Beziehung]

Integrierte Übung 3.6

Illustrieren Sie die Probleme die auftreten, wenn der 1:1 Beziehungstyp leiten durch einen Fremdschlüssel in der Relation Angestellte abgebildet wird, d.h., Angestellte die Rolle von S übernimmt.

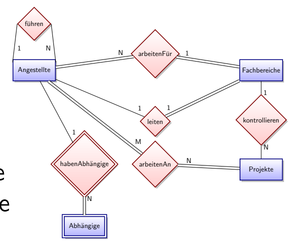
Schritt 4: Abbildung von 1:N Beziehungstypen

(a) Involvierte Entitätstypen: Für jeden 1:N Beziehungstyp identifizieren wir die Relationen T und S der involvierten Entitätstypen. S ist die N-Seite.
Beispiel: Fachbereiche kontrollieren Projekte

(b) Fremdschlüssel: Der Primärschlüssel von T wird als Fremdschlüssel zu S hinzugefügt.

(c) Attribute: Alle einfachen Attribute bzw. einfachen Komponenten zusammengesetzter Attribute des 1:N Beziehungstypen werden als Attribute zu S hinzugefügt.

Beispiel: Abbildung von 1:N Beziehungstypen



- **Beispiel:** Abbildung des N:1 Beziehungstyps Angestellte arbeitenFür Fachbereiche:
  - Angestellte entspricht der Relation S.
  - Primärschlüssel FNummer von Fachbereiche wird Fremdschlüssel der Relation Angestellte
- Weitere 1:N Beziehungstypen:
  - Angestellte/Vorgesetzte führen Angestellte/Unterstellte: Primärschlüssel von Angestellte als Fremdschlüssel VorgSVN zu Angestellte hinzufügen.
  - Fachbereiche kontrollieren Projekte: Primärschlüssel von Fachbereiche als Fremdschlüssel zu Projekte hinzufügen.

Angestellte[VName, NName, <u>SVN</u>, GDatum, Adresse, Geschlecht, Lohn, VorgSVN, FNummer]
Fachbereiche[FName, <u>FNummer</u>, LeiterSVN, StartDatum]
Projekte[PName, <u>PNummer</u>, PStandort, FNummer]
Abhängige[<u>AngSVN</u>, <u>AbhName</u>, Geschlecht, GDatum, Beziehung]

Schritt 5: Abbildung von M:N Beziehungstypen

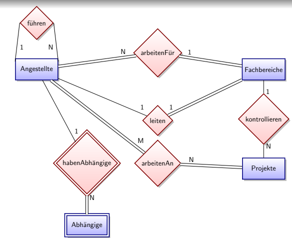
(a) Neue Relation: Für jeden M:N Beziehungstypen (z.B. Angestellte arbeitenAn Projekte) erstellen wir eine neue Relation R.

(b) Fremdschlüssel: Die Primärschlüssel der Relationen der involvierten Entitätstypen werden als Fremdschlüssel zu R hinzu.

(c) Primärschlüssel: Die Kombination der Fremdschlüssel ergibt den Primärschlüssel von R.

(d) Attribute: Alle einfachen Attribute bzw. einfachen Komponenten zusammengesetzter Attribute des M:N Beziehungstypen werden als Attribute zu R hinzugefügt.

Beispiel: Abbildung von M:N Beziehungstypen



- **Beispiel:** Für den M:N Beziehungstyp arbeitenAn erstellen wir eine Relation ArbeitenAn.
- Die Primärschlüssel der Relationen Projekte und Angestellte werden als Fremdschlüssel zur Relation ArbeitenAn hinzugefügt.
- Attribut *Stunden* der Relation ArbeitenAn bildet das gleichnamige Attribut des Beziehungstypen ab.
- Der Primärschlüssel der Relation ArbeitenAn ist die Kombination der Fremdschlüssel: { AngSVN, PNummer }.

Angestellte[VName, NName, SVN, GDatum, Adresse, Geschlecht, Lohn, VorgSVN, FNummer]
Fachbereiche[FName, FNummer, LeiterSVN, StartDatum]
Projekte[PName, PNummer, PStandort, FNummer]
Abhängige[AngSVN, AbhName, Geschlecht, GDatum, Beziehung]
ArbeitenAn[AngSVN, PNummer, Stunden]

Überblick: Abbildung von Beziehungen

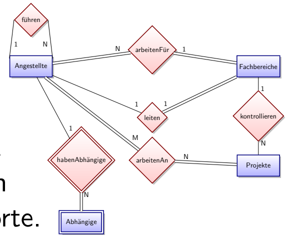
Drei mögliche Ansätze für Beziehung zwischen Entitätstypen S und T:

1. Zusammengefasste Relationen: nur für **1:1**
   - beteiligte Entitätstypen werden in einzige Relation zusammengelegt
   - keine Nullwerte falls S und T totale Beziehung eingehen
2. Fremdschlüssel: 1:1, **1:N**
   - eine der beteiligten Entitäten wird ausgewählt, z.B. S (N-Seite im Falle von 1:N)
   - Primärschlüssel von T wird als Fremdschlüssel zu S hinzugefügt
   - keine Nullwerte falls S totale Beziehung eingeht
3. Neue Beziehungsrelation: 1:1, 1:N, **M:N**
   - neue Relation R mit den Primärschlüsseln von S und T als Fremdschlüssel
   - Primärschlüssel:
     - $1 : 1$-Beziehung: einer der beiden Fremdschlüssel
     - $1 : N$-Beziehung: Fremdschlüssel der N-Seite
     - $M : N$-Beziehung: beide Fremdschlüssel
   - keine Nullwerte

Schritt 6: Abbildung mehrwertiger Attribute

- Neue Relation: Für jedes mehrwertige Attribut A erstellen wir eine neue Relation R.
- Attribute: Das mehrwertige Attribut A wird zur Relation R als (einfaches) Attribut hinzugefügt. Falls das mehrwertige Attribut A zusammengesetzt ist, werden alle einfachen Komponenten von A als (einfache) Attribute hinzugefügt.
- Fremdschlüssel: Primärschlüssel K der Relation, die den Entitäts- oder Beziehungstyp von A abbildet.
- Primärschlüssel: Kombination von A und K. Falls das mehrwertige Attribut zusammengesetzt ist, sind alle einfachen Komponenten Teil des Primärschlüssels.

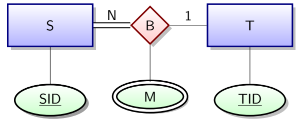
Beispiel 1: Abbildung mehrwertiger Attribute



- **Beispiel:** das mehrwertige Attribut Standorte des Entitätstyps Fachbereiche.
- Eine neue Relation FBStandorte mit Attribut Standort wird erstellt. FNummer der Relation Fachbereiche ist Fremdschlüssel in FBStandorte.
- Der Primärschlüssel von FBStandorte sind die Attribute { FNummer, Standort }.

Angestellte[VName, NName, <u>SVN</u>, GDatum, Adresse, Geschlecht, Lohn, VorgSVN, FNummer]
Fachbereiche[FName, <u>FNummer</u>, LeiterSVN, StartDatum]
Projekte[PName, <u>PNummer</u>, PStandort, FNummer]
Abhängige[<u>AngSVN</u>, <u>AbhName</u>, Geschlecht, GDatum, Beziehung]
ArbeitenAn[<u>AngSVN</u>, <u>PNummer</u>, Stunden]
FBStandorte[<u>FNummer</u>, <u>Standort</u>]

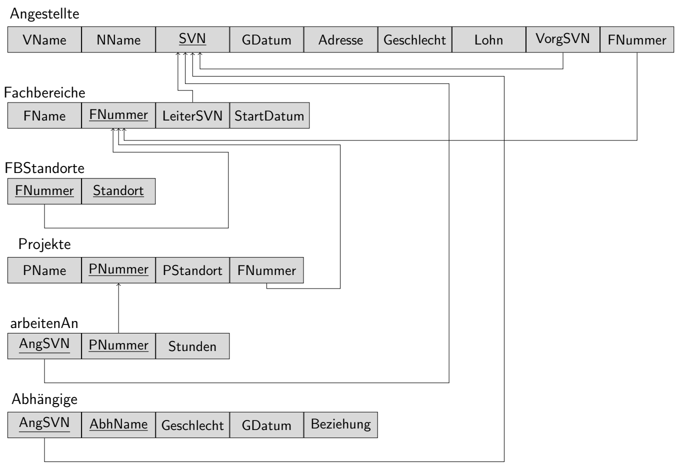
Beispiel 2: Abbildung mehrwertiger Attribute



- **Beispiel:** mehrwertiges Attribut M der 1:N Beziehung B
- 1:N Beziehung wird als Fremdschlüssel in *S* modelliert
- mehrwertiges Attribut wird durch neue Relation *MB* modelliert

S[<u>SID</u>, TID]
T[<u>TID</u>]
MB[<u>SID</u>,<u>M</u>]

Beispiel: Vollständige NAWI Datenbank

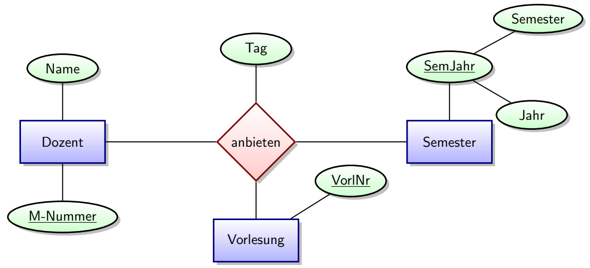


Schritt 7: Abbildung von n-wertigen Beziehungstypen.

- Neue Relation: Für jeden n-wertigen Beziehungstypen ($n > 2$) erstellen wir eine neue Relation R.
- Fremdschlüssel: Die Primärschlüssel der Relationen der involvierten Entitätstypen sind Fremdschlüssel in R.
- Primärschlüssel: Kombination aller Fremdschlüssel.
- Attribute: Alle einfachen Attribute bzw. einfachen Komponenten zusammengesetzter Attribute des M:N Beziehungstypen werden als Attribute zu R hinzugefügt.

Beispiel: Abbildung von n-wertigen Beziehungstypen.

- **Beispiel:** Der 3-wertige Beziehungstyp anbieten



- Der Beziehungstyp wird durch eine Relation Anbieten abgebildet. Der Primärschlüssel ist die Kombination der drei Fremdschlüssel: { M-Nummer, Jahr, Semester, VorlNr }

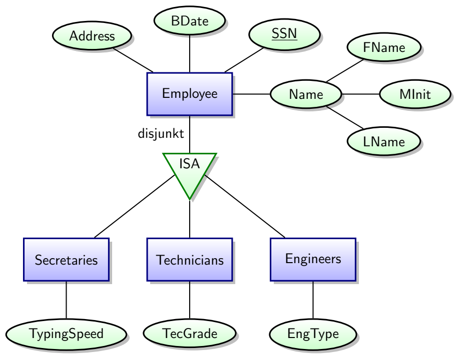
Dozent[M-Nummer, ...]
Semester[Jahr, Semester, ...]
Vorlesung[VorlNr, ...]
Anbieten[M-Nummer, Jahr, Semester, VorlNr, Tag, ...]

Schritt 8: Abbildung von Spezialisierung/Generalisierung

- Notation:
  - Untertyp: $U_1, U_2, \ldots, U_m$
  - Obertyp: O mit Attributen $k, a_1, a_2, \ldots, a_n$
  - k ist Primärschlüssel des Obertypen O
- Umsetztung:
  - Relation R für Obertyp O mit Attributen $attr(R) = \{\underline{k}, a_1, \ldots, a_n\}$.
  - Relation R_i für Untertypen U_i, $1 < i < m$, mit den Attributen $attr(R_i) = \{\,\underline{k}\,\} \cup \{$Attribute von $U_i\}$.
  - Attribute k der Relationen R_i sind Fremdschlüssel auf Attribut k in R.
- Kann für alle Arten der Spezialisierung verwendet werden:
  - vollständig und partiell
  - disjunkt und überlappend
- Einschränkung: vollständig und/oder disjunkt wird nicht erzwungen

Beispiel: Abbildung von Spezialisierung/1

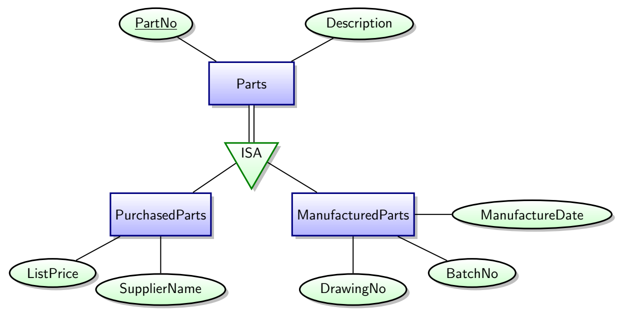
- **Beispiel:** Spezialisierung von Employee



Employee[SSN, FName, MInit, LName, BirthDate, Address, JobType]
Secretary[SSN, TypingSpeed]
Technician[SSN, TGrade]
Engineer[SSN, EngType]

Beispiel: Abbildung von Spezialisierung/2

- **Beispiel:** Spezialisierung von Parts



Parts[PartNo,Description]
PurchasedParts[PartNo,SupplierName,ListPrice]
ManufacturedParts[PartNo,DrawingNo,BatchNo,ManufactureDate]

Zusammenfassung der Abbildungen

Abbildung zwischen dem ER und dem relationalem Modell

ER Modell	**Relationales Modell**
Entitätstyp	Entitätsrelation
1:1 oder 1:N Beziehungstyp	Fremdschlüssel (oder Beziehungsrelation)
M:N Beziehungstyp	Beziehungsrelation mit 2 Fremdschlüsseln
n-wertige Beziehungstyp	Beziehungsrelation mit *n* Fremdschlüsseln
(Einfaches) Attribut	Attribut
zusammengesetztes Attribut	Menge von einfachen Attributen
Mehrwertiges Attribut	Relation mit Fremdschlüssel
Schlüsselattribut	Primärschlüssel
Spezialisierung	Relation für Ober- und Untertypen

Inhalt

Relationale Algebra

- Die relationale Algebra ist eine prozedurale Anfragesprache.
- Besteht aus sechs (notwendigen) Operatoren:
  - Selektion: σ
  - Projektion: π
  - Mengenvereinigung: $\cup$
  - Mengendifferenz: $-$
  - Kartesisches Produkt: $\times$
  - Umbenennung: ρ (Hilfsoperation)
- Die relationale Algebra ist abgeschlossen:
  - Argumente der Operatoren sind (ein oder zwei) Relationen.
  - Ergebnis der Operatoren ist wieder eine Relation.

Syntaktische Konventionen

- Es ist hilfreich bei der Namensgebung systematisch zu sein.
- Wir verwenden folgende Regeln.
  - Tabellennamen: Großschreibung und Plural
    Beispiele: Vorlesungen, Studenten, Module, R, S
  - Attributnamen: Großschreibung und Singular
    Beispiele: Semester, Jahr, Name, A, B
  - Konstanten (Werte):
    - Numerische Werte: 12, 17.6
    - Zeichenketten: durch Hochkommas begrenzen
      Beispiele: 'Martin', 'Mehr als ein Wort'
- Es gibt keinen einheitlichen Standard. Die Lehrbücher unterscheiden sich zum Teil.

Elementare Operatoren

- Selektion σ
- Projektion π
- Mengenvereinigung $\cup$
- Mengendifferenz $-$
- Kartesisches Produkt $\times$
- Umbenennung ρ

Selektion

- Notation: $\sigma_p(R)$ (sigma)
- Selektionsprädikat p ist aus folgenden Elementen aufgebaut:
  - Attributnamen der Argumentrelation R oder Konstanten als Operatoren
  - arithmetische Vergleichsoperatoren ($=, <, \leq, >, \geq$)
  - logische Operatoren: $\wedge$ (**and**), $\vee$ (**or**), $\neg$ (**not**)
- $p(t)$, $t \in R$ heißt: Prädikat p ist für Tupel t aus Relation R erfüllt.
- Definition: $t \in \sigma_p(R) \Leftrightarrow t \in R \wedge p(t)$
- Beispiel: $\sigma_{FiName='Brugg'}(Konten)$
- Beispiel: $\sigma_{A=B \wedge D>5}(R)$

R

A	B	C	D
α	α	1	7
α	β	5	7
β	β	12	3
β	β	23	10

$\sigma_{A=B \wedge D>5}(R)$

A	B	C	D
α	α	1	7
β	β	23	10

Projektion

- Notation: $\pi_{A_1,\ldots,A_k}(R)$ (pi)
- $A_1, A_2, \ldots, A_k$ sind Attribute von R und heißen Projektionsliste
- Definition: $t \in \pi_{A_1,\ldots,A_k}(R) \Leftrightarrow \exists x(x \in R \wedge t = x[A_1, \ldots, A_k])$, wobei $x[A_1, A_2, \ldots, A_k]$ ein neues Tupel bezeichnet, welches für die Werte von A_i, $1 \leq i \leq k$, die Werte der entsprechenden Attribute von x annimmt (alle Attribute A_i müssen in x vorkommen müssen)
- Beachte: Allfällige Duplikate (identische Tupel), die sich aus der Projektion ergeben, müssen entfernt werden.
- Beispiel: $\pi_{KoNr,Guthaben}(Konten)$
- Beispiel: $\pi_{A,C}(R)$

R

A	B	C
α	10	1
α	20	1
β	30	1
β	40	2

$\pi_{A,C}(R)$

A	C
α	1
β	1
β	2

Mengenvereinigung

- Notation: $R \cup S$
- Definition: $t \in (R \cup S) \Leftrightarrow t \in R \vee t \in S$
- $R \cup S$ ist nur definiert, wenn r und s das gleiche Schema haben (union compatible). Namensdifferenzen können durch explizites Umbenennung der Attribute eliminiert werden (s. weiter unten).
- Beispiel: $\pi_{KuName}(Kontoinhaber) \cup \pi_{KuName}(Kreditnehmer)$
- Beispiel: $R \cup S$

R

A	B
α	1
α	2
β	1

S

A	B
α	2
β	3

$R \cup S$

A	B
α	1
α	2
β	1
β	3

Mengendifferenz

- Notation: $R - S$
- Definition: $t \in (R - S) \Leftrightarrow t \in R \wedge t \notin S$
- Die Argumentrelationen der Mengendifferenz müssen das gleiche Schema haben (union compatible).
- Beispiel: $R - S$

R

A	B
α	1
α	2
β	1

S

A	B
α	2
β	3

$R - S$

A	B
α	1
β	1

Kartesisches Produkt (Kreuzprodukt)

- Notation: $R \times S$
- Definition: $t \in (R \times S) \Leftrightarrow \exists x, y (x \in R \wedge y \in S \wedge t = x \circ y)$
- $\circ$ bezeichnet die Konkatenation von Tupeln: $[1,2] \circ [5] = [1,2,5]$
- Die Attribute von R und S müssen unterschiedliche Namen haben.
- Beispiel: $R \times S$

R

A	B
α	1
β	2

S

C	D	E
α	10	a
β	10	a
β	20	b
γ	10	b

$R \times S$

A	B	C	D	E
α	1	α	10	a
α	1	β	10	a
α	1	β	20	b
α	1	γ	10	b
β	2	α	10	a
β	2	β	10	a
β	2	β	20	b
β	2	γ	10	b

Umbenennung

- Erlaubt es den Namen der Relation und der Attribute eines algebraischen Ausdrucks E zu spezifizieren.
- Wird auch verwendet um Namenskonflikte aufzulösen (z.B., in Mengenvereinigung oder Kreuzprodukt)
- Verschiedene Variationen (E ist ein relationaler Ausdruck):
  - $\rho_R(E)$ ist eine Relation mit Namen R.
  - $\rho_{R[A_1,\ldots,A_k]}(E)$ ist eine Relation mit Namen R und Attributnamen $A_1, \ldots, A_k$.
  - $\rho_{[A_1,\ldots,A_k]}(E)$ ist eine Relation mit Attributnamen $A_1, \ldots, A_k$.
- Beispiel: $\rho_{S[X,Y,U,V]}(R)$

R

A	B	C	D
α	α	1	7
β	β	23	10

S

X	Y	U	V
α	α	1	7
β	β	23	10

Zusammengesetzte Ausdrücke

- Geschachtelte Ausdrücke: Da die relationale Algebra abgeschlossen ist, d.h. das Resultat eines Operators der relationalen Algebra ist wieder eine Relation, ist es möglich Ausdrücke zu schachteln.
- Beispiel: $\sigma_{A=C}(R \times S)$

R

A	B
α	1
β	2

S

C	D	E
α	10	a
β	10	a
β	20	b
γ	10	b

$R \times S$

A	B	C	D	E
α	1	α	10	a
α	1	β	10	a
α	1	β	20	b
α	1	γ	10	b
β	2	α	10	a
β	2	β	10	a
β	2	β	20	b
β	2	γ	10	b

$\sigma_{A=C}(R \times S)$

A	B	C	D	E
α	1	α	10	a
β	2	β	10	a
β	2	β	20	b

Integrierte Übung 3.7

- Identifizieren und korrigieren Sie Fehler in den nachfolgenden relationalen Algebra Ausdrücken. Relation R hat Schema $sch(R) = [A, B]$.
- $\sigma_{R.A>5}(R)$
- $\sigma_{A,B}(R)$
- $R \times R$

Integrierte Übung 3.8

- Identifizieren und korrigieren Sie Fehler in den nachfolgenden relationalen Algebra Ausdrücken. Relation $Pers$ hat Schema $sch(Pers) = [Name, Alter, Stadt]$.
- $\sigma_{Name='Name'}(Pers)$
- $\sigma_{Stadt=Zuerich}(Pers)$
- $\sigma_{Alter>'20'}$

Beispiel: Banken

Filialen[FiName, Stadt, Umsatz]
Kunden[KuName, Strasse, Ort]
Konten[KoNr, FiName, Guthaben]
Kredite[KrNr, FiName, Betrag]
Kontoinhaber[KuName, KoNr]
Kreditnehmer[KuName, KrNo]

Fremdschlüssel:

- $\pi_{\text{FiName}}(\text{Konten}) \subseteq \pi_{\text{FiName}}(\text{Filialen})$
- $\pi_{\text{FiName}}(\text{Kredite}) \subseteq \pi_{\text{FiName}}(\text{Filialen})$
- $\pi_{\text{KuName}}(\text{Kontoinhaber}) \subseteq \pi_{\text{KuName}}(\text{Kunden})$
- $\pi_{\text{KoNr}}(\text{Kontoinhaber}) \subseteq \pi_{\text{KoNr}}(\text{Konten})$
- $\pi_{\text{KuName}}(\text{Kreditnehmer}) \subseteq \pi_{\text{KuName}}(\text{Kunden})$
- $\pi_{\text{KoNo}}(\text{Kreditnehmer}) \subseteq \pi_{\text{KrNr}}(\text{Kredite})$

Anfragebeispiele/1

Filialen[FiName, Stadt, Umsatz]
Kunden[KuName, Strasse, Ort]
Konten[KoNr, FiName, Guthaben]
Kredite[KrNr, FiName, Betrag]
Kontoinhaber[KuName, KoNr]
Kreditnehmer[KuName, KrNo]

- Jene Kredite die größer als \$1200 sind.

  $\sigma_{Betrag>1200}(Kredite)$

- Die Nummern jener Kredite die größer als \$1200 sind.

  $\pi_{KrNr}(\sigma_{Betrag>1200}(Kredite))$

- Die Namen aller Kunden die einen Kredit oder ein Konto (oder beides) haben.

  $\pi_{KuName}(Kreditnehmer) \cup \pi_{KuName}(Kontoinhaber)$

Anfragebeispiele/2

Filialen[FiName, Stadt, Umsatz]
Kunden[KuName, Strasse, Ort]
Konten[KoNr, FiName, Guthaben]
Kredite[KrNr, FiName, Betrag]
Kontoinhaber[KuName, KoNr]
Kreditnehmer[KuName, KrNo]

- Die Namen aller Kunden die einen Kredit bei der Brugg Filiale haben.
  - Anfrage 1

$$\pi_{KuName}(\sigma_{FiName='Brugg'}(\sigma_{KrNo=KrNr}(Kreditnehmer \times Kredite)))$$

  - Anfrage 2

$$\pi_{KuName}(\sigma_{KrNo=KrNr}((\sigma_{FiName='Brugg'}(Kredite)) \times Kreditnehmer))$$

Anfragebeispiele/3

Filialen[FiName, Stadt, Umsatz]
Kunden[KuName, Strasse, Ort]
Konten[KoNr, FiName, Guthaben]
Kredite[KrNr, FiName, Betrag]
Kontoinhaber[KuName, KoNr]
Kreditnehmer[KuName, KrNo]

- Die Namen aller Kunden die einen Kredit bei der Brugg Filiale haben, aber kein Konto bei der Bank.

$$\pi_{KuName}(\sigma_{FiName='Brugg'}(\sigma_{KrNo=KrNr}(Kreditnehmer \times Kredite))) - \pi_{KuName}(Kontoinhaber)$$

Integrierte Übung 3.9

- Gegeben: Relation $R[A] = \{[1], [2], [3]\}$. Schreiben Sie einen relationalen Algebra Ausdruck der den größten Wert in R bestimmt.

Anfragebeispiele/4

Filialen[FiName, Stadt, Umsatz]
Kunden[KuName, Strasse, Ort]
Konten[KoNr, FiName, Guthaben]
Kredite[KrNr, FiName, Betrag]
Kontoinhaber[KuName, KoNr]
Kreditnehmer[KuName, KrNo]

- Das Konto (bzw. die Konten) mit dem höchsten Kontostand.
- Lösungsidee:
  - Bestimmen jener Konten die **nicht** den höchsten Kontostand haben (indem man jedes Konto mit allen anderen Konten vergleicht)
  - Mit Hilfe der Mengendifferenz werden jene Konten bestimmt die im ersten Schritt nicht gefunden wurden.
- Lösung:

$$\pi_{KoNr}(Konten) - \pi_{KoNr}(\sigma_{Guthaben<Guth}(Konten \times \rho_{[Nr,Fil,Guth]}(Konten)))$$

Integrierte Übung 3.10

- Gegeben eine Relation
  Verbindet[VonBhf, NachBhf, ZugNr, Abfahrt, Ankunft]
- Beispiel einer Instanz:

Verbindet

VonBhf	NachBhf	ZugNr	Abfahrt	Ankunft
Zurich	Lenzburg	IC 706	5:21	5:40
Lenzburg	Aarau	IC 706	5:40	5:47
Aarau	Olten	IC 706	5:49	5:58
Zurich	Olten	IC 812	17:24	17:58
Olten	Biel	IC 812	18:01	18:34
Zurich	Lenzburg	IR 1798	0:08	0:27
...	...	...	...	...

- Bestimmen Sie alle direkten Zugverbindungen (d.h. ohne Umsteigen) von Zürich nach Olten. Annahme: Keine Züge verkehren über Mitternacht.

Definition von relationalen Algebra Ausdrücken

- Ein elementarer Ausdruck der relationalen Algebra ist eine Relation in der Datenbank (z.B. Konten).
- Falls E_1 und E_2 relationale Algebra Ausdrücke sind, dann lassen sich weitere relationale Algebra Ausdrücke wie folgt bilden:
  - $E_1 \cup E_2$
  - $E_1 - E_2$
  - $E_1 \times E_2$
  - $\sigma_p(E_1)$, p ist ein Prädikat in E_1
  - $\pi_s(E_1)$, s ist eine Liste mit Attributen aus E_1
  - $\rho_x(E_1)$, x ist der Name für E_1

Notationsvarianten der Relationalen Algebra

- Im Laufe der Zeit sind unterschiedliche Notationen entstanden.
- Notation von Kemper&Eikler (Lehrbuch) unterscheidet sich wie folgt.
- Qualifizierte Attributnamen
  - Attributnamen werden durch Voranstellen des Relationsnamen eindeutig gemacht (wo nötig), z.B., $R.B$, $S.B$
  - Kreuzprodukt $R \times S$ ist auch dann erlaubt, wenn R und S gleichnamige Attribute haben
  - Beispiele: Gegeben $R[A, B]$, $S[B, C]$
    - $sch(R \times S) = [A, R.B, S.B, C]$
    - $\sigma_{R.B=S.B}(R \times S)$ ist syntaktisch korrekt
- Umbenennung mit Zuordnung
  - Syntax von ρ unterscheidet sich für Relationen und Attribute
  - Relation: $\rho_R(E)$ benennt relationalen Ausdruck E mit R
  - Attribut: $\rho_{A \leftarrow B}(R)$ benennt Attribut A in B um ($A \in sch(R)$)

In der Prüfung ist die Notation aus der Vorlesung zu verwenden.

Zusammenfassung: Elementare Operatoren

- Relationale Algebra ist prozedural und abgeschlossen.
- Elementare Operatoren:
  - unär: Selektion σ, Projektion π, Umbenennung ρ
  - binär: Mengenvereinigung $\cup$, Mengendifferenz $-$, Kreuzprodukt $\times$
- Ein relationaler Ausdruck kann sein:
  - ein elementarer Ausdruck (Relation)
  - eine Kombination von relationalen Ausdrücken, die über relationale Operatoren verbunden sein müssen

Zusätzliche Operatoren der Relationalen Algebra

- Neben den elementaren Operatoren gibt es zusätzliche Operatoren:
  - Mengendurchschnitt $\cap$
  - Join $\bowtie$
  - Zuweisung $\leftarrow$
- Die zusätzlichen Operatoren machen Algebra nicht ausdrucksstärker:
  - man kann die zusätzlichen Operatoren mithilfe der elementaren Operatoren ausdrücken
  - deshalb sind die zusätzlichen Operatoren *redundant*
- Formulierung häufiger Anfragen wird zum Teil erheblich vereinfacht.

Mengendurchschnitt

- Notation: $R \cap S$
- Definition: $t \in (R \cap S) \Leftrightarrow t \in R \wedge t \in S$
- Voraussetzung: R und S haben das gleiche Schema
- Beachte: $R \cap S = R - (R - S)$
- Beispiel: $R \cap S$

R

A	B
α	1
α	2
β	1

S

A	B
α	2
β	3

$R \cap S$

A	B
α	2

Theta Join (Verbund)/1

- Notation: $R \bowtie_\theta S$
- Annahme: R und S sind Relationen. θ ist ein Prädikat über den Attributen von R und S.
- $R \bowtie_\theta S$ ist eine Relation mit einem Schema das aus allen Attributen von $sch(R)$ und allen Attributen von $sch(S)$ besteht.
- Beispiel:
  - $sch(R) = [A, B, D]$ und $sch(S) = [X, Y, Z]$
  - $R \bowtie_{A=Z} S$
  - Schema des Resultats ist $[A, B, D, X, Y, Z]$
  - Äquivalent zu: $\sigma_{A=Z}(R \times S)$

R

A	B	D
α	1	a
β	2	a
γ	4	b

S

X	Y	Z
1	a	α
3	a	β
3	b	ϵ

$\sigma_{A=Z}(R \times S)$

A	B	D	X	Y	Z
α	1	a	1	a	α
β	2	a	3	a	β

Theta Join (Verbund)/2

- Beispiel:
  - $sch(R) = [A, B, D]$ und $sch(S) = [X, Y, Z]$
  - $R \bowtie_{A=Z \wedge B<X} S$
  - Schema des Resultats ist $[A, B, D, X, Y, Z]$
  - Äquivalent zu: $\sigma_{A=Z \wedge B<X}(R \times S)$

R

A	B	D
α	1	a
β	2	a
γ	4	b

S

X	Y	Z
1	a	α
3	a	β
3	b	ϵ

$\sigma_{A=Z}(R \times S)$

A	B	D	X	Y	Z
β	2	a	3	a	β

Natürlicher Join

- Notation: $R \bowtie S$
- Annahme: R und S sind Relationen.
- Der natürliche Join verlangt, dass Attribute die sowohl in R als auch S vorkommen identische Werte haben.
- Das Resultat von $R \bowtie S$ ist eine Relation mit einem Schema das alle Attribute von R enthält und alle Attribute von S die nicht in R vorkommen.
- Beispiel:
  - $R \bowtie S$ mit $sch(R) = [A, B, D]$ und $sch(S) = [B, D, E]$
  - Schema des Resultats ist $[A, B, D, E]$
  - Äquivalent zu: $\pi_{A,B,D,E}(\sigma_{B=Y \wedge D=Z}(R \times \rho_{[Y,Z,E]}(S))$

R

A	B	D
α	1	a
β	2	a

S

B	D	E
1	a	α
3	a	β

$R \bowtie S$

A	B	D	E
α	1	a	α

Semi- und Anti-Join

- Semi-Join: $R \ltimes S$
  - alle Tupel von R die in einem natürlichen Join mit S mindestens einen Join-Partner finden.
  - $R \ltimes S = \pi_{sch(R)}(R \bowtie S)$
- Anti-Join: $R \rhd S$
  - alle Tupel von R die in einem natürlichen Join mit S keinen Join-Partner finden.
  - $R \rhd S = R - (R \ltimes S)$

Zuweisung

- Die Zuweisung ($\leftarrow$) erlaubt es, komplexe Ausdrücke in kleinere übersichtliche Blöcke aufzubrechen.
  - links von $\leftarrow$ steht eine Variable
  - rechts von $\leftarrow$ steht ein relationaler Algebra Ausdruck
  - das Resultat rechts von $\leftarrow$ wird der Variablen links von $\leftarrow$ zugewiesen
  - komplexe Ausdrücke werden als Sequenz von Zuweisungen geschrieben
- Beispiel: Das Konto mit dem höchsten Kontostand (s.o.) kann wie folgt geschrieben werden:

$$Tmp1 \leftarrow \pi_{KoNr}(Konten)$$
$$Tmp2 \leftarrow \pi_{KoNr}(\sigma_{Guthaben<Guth}(Konten \times \rho_{[Nr,Fil,Guth]}(Konten)))$$
$$Result \leftarrow Tmp1 - Tmp2$$

Bankbeispiel Anfragen/1

Filialen[FiName, Stadt, Umsatz]
Kunden[KuName, Strasse, Ort]
Konten[KoNr, FiName, Guthaben]
Kredite[KrNr, FiName, Betrag]
Kontoinhaber[KuName, KoNr]
Kreditnehmer[KuName, KrNo]

- Alle Kunden die sowohl ein Konto als auch einen Kredit haben.

$$\pi_{KuName}(Kreditnehmer) \cap \pi_{KuName}(Kontoinhaber)$$

- Name und Kreditbetrag aller Kunden die einen Kredit haben.

Lösung 1: $\pi_{KuName,Betrag}(Kreditnehmer \bowtie_{KrNo=KrNr} Kredite)$

Lösung 2: $\pi_{KuName,Betrag}(\rho_{[KuName,KrNr]}(Kreditnehmer) \bowtie Kredite)$

Bankbeispiel Anfragen/2

Filialen[FiName, Stadt, Umsatz]
Kunden[KuName, Strasse, Ort]
Konten[KoNr, FiName, Guthaben]
Kredite[KrNr, FiName, Betrag]
Kontoinhaber[KuName, KoNr]
Kreditnehmer[KuName, KrNo]

- Kunden die sowohl ein Konto bei der Filiale Chur als auch der Filiale Lanquart haben.
  - Lösung:

$$\pi_{KuName}(\sigma_{FiName='Chur'}(Kontoinhaber \bowtie Konten))$$
$$\cap$$
$$\pi_{KuName}(\sigma_{FiName='Lanquart'}(Kontoinhaber \bowtie Konten))$$

Zusammenfassung: Zusätzliche Operatoren

- Zusätzliche Operatoren der relationalen Algebra:
  - Mengendurschnitt $\cap$
  - Join (theta, natural) $\bowtie$
  - Zuweisung $\leftarrow$
- Zusätzliche Operatoren verändern nicht die Ausdrucksstärke der relationalen Algebra, vereinfachen aber die Anfragen.
- Besonders der Join Operator spielt eine große Rolle in der effizienten Implementierung der relationalen Algebra in Systemen.

Operatoren der Erweiterten Relationalen Algebra

Die erweiterten Operatoren erhöhen die Ausdrucksstärke der relationalen Algebra.

- Verallgemeinerte Projektion π
- Gruppierung und Aggregation γ
- Äußerer Join (outer join) ⟕, ⟖, ⟗

Verallgemeinerte Projektion

- Erlaubt arithmetische Funktionen in der Projektionsliste:

$$\pi_{F_1,F_2,\ldots,F_n}(E)$$

- E ist ein relationaler Ausdruck.
- $F_1, F_2, \ldots, F_n$ sind jeweils arithmetische Ausdrücke, welche Konstanten und Attribute des Schemas von E enthalten.
- Beispiel: Gegeben eine Relation $Kredite[Kunde, Limit, KreditBetrag]$, finde heraus, wieviel jeder Kunde noch ausgeben darf:

$$\pi_{Kunde,Limit-KreditBetrag}(Kredite)$$

Aggregationsfunktionen

- Aggregationsfunktionen erhalten eine Multimenge von Werten als Argument und liefern als Ergebnis einen einzigen Funktionswert.
  - avg: Durchschnitt
  - min: kleinster Wert
  - max: größter Wert
  - sum: Summe aller Werte
  - count: Anzahl der Werte (Kardinalität der Menge/Multimenge)
- Elemente der Argumentmenge und Funktionswert sind atomar, nicht Tupel.
- Multimenge (Menge mit Duplikaten): k-fache Werte gehen k-fach in die Berechnung ein.
- Beispiele: ($\{\ldots\}_m$ ist eine Multimenge)
  - $\text{min}(\{3, 1, 5, 5\}_m) = 1$
  - $\text{count}(\{3, 1, 5, 5\}_m) = 4$
  - $\text{avg}(\{3, 1, 5, 5\}_m) = 3.5$

Gruppierung

- Partitionierung der Tupel einer Relation gemäß ihrer Werte in einem oder mehreren Attributen.
- Gruppe (Parition): Alle Tupel mit identischen Werten in allen Gruppierungsattributen.
- Hauptzweck: Aggregation auf Teilen einer Relation (Gruppen)
- Beispiel: Gegeben Relation $R = \{[1, 2, 3], [1, 2, 5], [1, 4, 3], [2, 3, 5], [2, 4, 5]\}$ mit Schema $sch(R) = [A, B, C]$.
  - Gruppierung nach Attribut A ergibt die Gruppen $\{[1, 2, 3], [1, 2, 5], [1, 4, 3]\}$ und $\{[2, 3, 5], [2, 4, 5]\}$
  - Gruppierung nach den Attributen A, C ergibt die Gruppen $\{[1, 2, 3], [1, 4, 3]\}, \{[1, 2, 5]\}, \{[2, 3, 5], [2, 4, 5]\}$

Gruppierungsoperator

- Die Gruppierung der relationalen Algebra:

  $\gamma_{G_1, G_2, \ldots, G_m; F_1(A_1), F_2(A_2), \ldots, F_n(A_n)}(R)$

  R ist eine Relation:
  - Gruppierungsattribute: $G_1, G_2 \ldots, G_m$ ist eine Liste von Attributen aus R, über die gruppiert wird (kann leer sein)
  - Aggregationsfunktionen: F_i ist eine Aggregationsfunktion
  - Aggregierte Attribute: A_i ist ein Attribut von R
- Leere Attributeliste: Gruppe besteht aus der gesamten Relation R.
- Ergebnis: Relation mit $m + n$ Attributen
  - Anzahl der Tupel entspricht Anzahl der Gruppen (ein Tupel pro Gruppe)
  - die Werte der ersten m Attribute des Tupels einer Gruppe entsprechen $G_1, G_2, \ldots, G_m$ (Wert gleich für alle Tupel in der Gruppe)
  - die letzten n Attribute entsprechen den Funktionsergebnissen von F_i über die (Multi-)menge aller Werte von A_i in der Gruppe

Beispiel: Gruppierungsoperator

- Relation R, $Res \leftarrow \rho_{[SumC]}(\gamma_{sum(C)}(R))$

r

A	B	C
α	α	7
α	β	7
β	β	3
β	β	10

Res

sumC
27

- Gesamteinlagen pro Filiale:
  $Res \leftarrow \rho_{[FiName, SumEinlagen]}(\gamma_{FiName; sum(Guthaben)}(Konten))$

Konten

FiName	*KoNr*	*Guthaben*
Perryridge	A-102	400
Perryridge	A-201	900
Brighton	A-217	750
Brighton	A-215	750
Redwood	A-222	700

Res

FiName	*SumEinlagen*
Perryridge	1300
Brighton	1500
Redwood	700

Äußerer Join (Outer Join)

- Erweiterung des Join Operators, welche Informationsverlust verhindert.
- Berechnet Join und fügt die Tupel, die keinen Join-Partner haben, zum Join-Ergebnis hinzu.
- Varianten:
  - (Voller) äußerer Join ($R ⟗ S$): erhält Tupel von R und S
  - Linker äußerer Join ($R ⟕ S$): erhält nur Tupel von R (linke Relation)
  - Rechter äußerer Join ($R ⟖ S$): erhält nur Tupel von S (rechte Relation)
- Verwendet `null` Werte, um die neuen Attribute der Tupel ohne Join-Partner zu füllen.
- Analog zum "normalen" (inneren) Join gibt es einen natürlichen und einen theta äußeren Join.

Beispiel: Äußerer Join/1

- Beispiel Relationen:

Kredite

KrNr	*FiName*	*Betrag*
L-170	Downtown	3000
L-230	Redwood	4000
L-260	Perryridge	1700

Kreditnehmer

KuName	*KrNr*
Jones	L-170
Smith	L-230
Hayes	L-155

- Join (auch "innerer" Join genannt)

Kredite ⋈ *Kreditnehmer*

KrNr	*FiName*	*Betrag*	*KuName*
L-170	Downtown	3000	Jones
L-230	Redwood	4000	Smith

Beispiel: Äußerer Join/2

- Beispiel Relationen:

Kredite

KrNr	*FiName*	*Betrag*
L-170	Downtown	3000
L-230	Redwood	4000
L-260	Perryridge	1700

Kreditnehmer

KuName	*KrNr*
Jones	L-170
Smith	L-230
Hayes	L-155

- Linker äußerer Join (erhält Tupel der linken Relation)

Kredite ⟕ *Kreditnehmer*

KrNr	*FiName*	*Betrag*	*KuName*
L-170	Downtown	3000	Jones
L-230	Redwood	4000	Smith
L-260	Perryridge	1700	*null*

Beispiel: Äußerer Join/3

- Beispiel Relationen:

Kredite

KrNr	*FiName*	*Betrag*
L-170	Downtown	3000
L-230	Redwood	4000
L-260	Perryridge	1700

Kreditnehmer

KuName	*KrNr*
Jones	L-170
Smith	L-230
Hayes	L-155

- Rechter äußerer Join (erhält Tupel der rechten Relation)

Kredite ⟖ *Kreditnehmer*

KrNr	*FiName*	*Betrag*	*KuName*
L-170	Downtown	3000	Jones
L-230	Redwood	4000	Smith
L-155	*null*	*null*	Hayes

Beispiel: Äußerer Join/4

- Beispiel Relationen:

Kredite

KrNr	*FiName*	*Betrag*
L-170	Downtown	3000
L-230	Redwood	4000
L-260	Perryridge	1700

Kreditnehmer

KuName	*KrNr*
Jones	L-170
Smith	L-230
Hayes	L-155

- (Vollständiger) äußerer Join (erhält Tupel beider Relationen)

$Kredite ⟗ Kreditnehmer$

KrNr	*FiName*	*Betrag*	*KuName*
L-170	Downtown	3000	Jones
L-230	Redwood	4000	Smith
L-260	Perryridge	1700	*null*
L-155	*null*	*null*	Hayes

Zusammenfassung: Erweiterte Relationale Algebra

- Erweiterte Relationale Algebra ist ausdrucksstärker als elementare relationale Algebra.
- Verallgemeinerte Projektion π: Arithmetik in Projektionsliste
- Gruppierung und Aggregation ϑ: Berechnung über Gruppen von Attributwerten
- Äußerer Join ⟖, ⟕, ⟗: Tupel-erhaltender Join

Änderung der Datenbank

- Der Inhalt der Datenbank kann mithilfe folgenden Operatoren verändert werden:
  - Löschen (delete)
  - Einfügen (insert)
  - Ändern (update)
- All diese Operationen verwenden den Zuweisungsoperator.

Löschen

- Ausdruck ähnlich einer Anfrage, wobei die Ergebnistupel von der Datenbank entfernt werden.
- Nur ganze Tupel können entfernt werden; Werte einzelner Attribute können nicht entfernt werden.
- Löschen wird in der relationalen Algebra folgendermaßen ausgedrückt:

$$R \leftarrow R - E$$

wobei R eine Relation ist und E ein Ausdruck der relationalen Algebra.

Beispiel: Löschen

Filialen[FiName, Stadt, Umsatz]
Kunden[KuName, Strasse, Ort]
Konten[KoNr, FiName, Guthaben]
Kredite[KrNr, FiName, Betrag]
Kontoinhaber[KuName, KoNr]
Kreditnehmer[KuName, KrNo]

- Lösche alle Konten in der Filiale Domplatz:

$$R_1 \leftarrow \sigma_{FiName='Domplatz'}(Konten)$$
$$Konten \leftarrow Konten - R_1$$
$$R_2 \leftarrow \pi_{KuName,KoNr}(R_1 \bowtie Kontoinhaber)$$
$$Kontoinhaber \leftarrow Kontoinhaber - R_2$$

Einfügen

- Es gibt zwei Möglichkeiten, um Daten in die Relation einzufügen:
  - die einzufügenden Tupel explizit angeben
  - eine Anfrage schreiben deren Ergebnis eingefügt werden soll
- Einfügen wird in der relationalen Algebra folgendermaßen ausgedrückt:

$$R \leftarrow R \cup E$$

  wobei R eine Relation und E ein relationaler Ausdruck sind.
- Wird ein einzelnes, explizites Tupel eingefügt, ist E eine konstante Relation die nur ein Tupel enthält.

Beispiel: Einfügen/1

Filialen[FiName, Stadt, Umsatz]
Kunden[KuName, Strasse, Ort]
Konten[KoNr, FiName, Guthaben]
Kredite[KrNr, FiName, Betrag]
Kontoinhaber[KuName, KoNr]
Kreditnehmer[KuName, KrNo]

- Füge folgende Information in die Datenbank ein: Kunde Smith eröffnet ein neues Konto mit Nummer A-973 auf der Domplatz Filiale und legt 1200 EUR ein.

$$Konten \leftarrow Konten \cup \{[\text{`A-973`}, \text{`}Domplatz\text{`}, 1200]\}$$
$$Kontoinhaber \leftarrow Kontoinhaber \cup \{[\text{`}Smith\text{`}, \text{`A-973`}]\}$$

Beispiel: Einfügen/2

Filialen[FiName, Stadt, Umsatz]
Kunden[KuName, Strasse, Ort]
Konten[KoNr, FiName, Guthaben]
Kredite[KrNr, FiName, Betrag]
Kontoinhaber[KuName, KoNr]
Kreditnehmer[KuName, KrNo]

- Alle Kreditnehmer der Domplatz Filiale erhalten ein Konto mit 200 EUR Guthaben geschenkt, wobei die Kontonummer des neuen Kontos identisch mit der jeweiligen Kreditnummer ist.

$$R_1 \leftarrow \sigma_{FiName='Domplatz'}(Kreditnehmer \bowtie_{KrNo=KrNr} Kredite)$$
$$Konten \leftarrow Konten \cup \rho_{KoNr,FiName,Guthaben}(\pi_{KrNr,FiName}(R_1) \times \{[200]\})$$
$$Kontoinhaber \leftarrow Kontoinhaber \cup \rho_{KuName,KoNr}(\pi_{KuName,KrNr}(R_1))$$

Änderung

- Änderungen erlauben, einzelne Werte eines Tupels zu ändern, ohne alle Werte ändern zu müssen.
- Kann durch Löschen und Einfügen ausgedrückt werden.
  - in realen Systemen ist die Änderungsoperation jedoch oft viel schneller
  - deshalb gibt es einen eigenen Operator
- In relationaler Algebra werden Änderungen in der Relation R durch Ersetzten der Relation R durch einen relationalen Ausdruck E ausgedrückt:

  $R \leftarrow E$
- Oft ist E eine erweiterte Projektion über $R[A_1, A_2, \ldots, A_n]$:

  $R \leftarrow \pi_{F_1, F_2, \ldots, F_n}(R)$

  wobei F_i
  - entweder A_i ist, falls Attribut A_i nicht geändert werden soll
  - oder eine Funktion, die einen neuen Wert für A_i festlegt.

Beispiel: Änderung

- Auszahlung der Zinsen von 5% auf alle Konten:

  $Konten \leftarrow \pi_{KoNr, FiName, Guthaben*1.05}(Konten)$
- Zahle 6% Zinsen für alle Konten mit mehr als 10.000 EUR Guthaben und 5% für alle anderen Konten:

  $Konten \leftarrow$

  $\pi_{KoNr, FiName, Guthaben*1.06}(\sigma_{Guthaben>10000}(Konten))$

  $\cup$

  $\pi_{KoNr, FiName, Guthaben*1.05}(\sigma_{Guthaben\leq 10000}(Konten))$

Zuasammenfassung

- Relationale Manipulationssprache
  - Löschen, Einfügen, Ändern
  - Wird durch Zuweisungsoperator ($\leftarrow$) und Ausdrücken der relationalen Algebra ausgedrückt.

Zuasammenfassung

Relationale Algebra:

- Elementare Operatoren: notwendig
- Zusätzliche Operatoren: redundant (können durch elementare Operatoren ausgedrückt werden)
- Erweiterte Operatoren: erhöhen die Ausdrucksstärke
- Manipulationssprache: Zuweisungsoperator und relationale Algebra