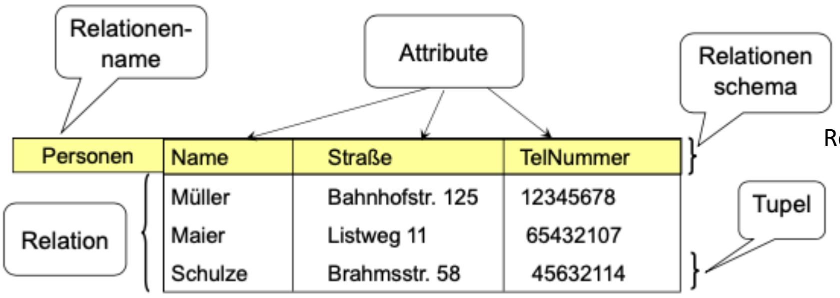
Kapitel 3 Implementierungsmodell

Relationen Modell

- Das relationale Modell definiert eine Datenstruktur
  - Alle Informationen ausschließlich durch atomare Werte dargestellt
- Implementierung von Entitätstypen und Beziehungstypen durch Relationen
- Definiert Operationen auf (mehreren) Relationen
  - Vereinigung, Durchschnitt, Differenz, Kartesisches Produkt
  - Projektion, Selektion, Verbundoperationen

Datenstrukturen



Relationenschema Personen(Name, Straße, TelNummer)
dom(Name) = dom(Strasse) = {Zeichenkette}
dom(TelNummer) = {8 - stellige Zahl}

  - Sortierung der Zeilen ohne Bedeutung
  - Sortierung der Spalten ohne Bedeutung (eindeutiger Attributname)
  - Alle Werte sind atomar

Relation und Relationen Schema

Gegeben seien n Wertebereiche (Domänen) $D_1, D_2, ..., D_n$, die nur atomare Werte enthalten

- atomare Werte: Zeichenketten (Strings), Zahlen

Eine **Relation** r ist definiert als Teilmenge des kartesischen Produktes (Kreuzprodukt) der Domänen

$$r \subseteq D_1 \times D_2 \times ... \times D_n$$

Daten der realen Welt werden als Sammlung von Relationen dargestellt:

- Relation = Tabelle
- Spalte = Werte aus einem Wertebereich D_i
- Zeile = Tupel $(d_1, d_2, ..., d_n)$

mathematisch:

- n: Anzahl der Attribute (**Grad** der Relation)
- D_i: Wertebereich des Attributes Ai ist eine Menge
- $r \in D_1 \times D_2 \times ... \times D_n$: **n - stellige Relation** über den Wertebereichen $D_1, ..., D_n$
- $(d_1, d_2, ..., d_n) \in r$: **Tupel** der Relation R
- $d_i \in D_i$: i - te Komponente des Tupels

Zusammenfassung:

- Relationenschema R $(A_1, A_2, ..., A_n)$:
  - R ist der Relationenname
  - A_i ist der Name des i-ten Attributes
  - Jedem Attribut A_i ist ein Wertebereich $dom(A_i) = D_i$ zugeordnet.
- Zu jedem Zeitpunkt existiert genau eine Relation r zum Relationenschema R: "r ist eine Instanz von R".
- Die zu R gehörigen Relationen sind also sämtliche Relationen des Typs
  $r \in dom(A_1) \times dom(A_2) \times ... \times dom(A_n) = D_1 \times ... \times D_n$
- Die Gesamtheit der Relationenschemata einer Datenbank
  heißt Schema der relationalen Datenbank oder **Datenbankschema**
  - Zusätzlich gehören zum Schema noch **Integritätsbedingungen**

Beispiel: Kunde(Kundennr, Name, Telefon)
dom(Kundennr) = int || dom(Name) = String || dom(Telefon) = Menge der Zeichenketten

Kunde = { (4711, „Mustermann“, „+49 511 47 114 711“),
(4712, „Musterfrau“, „+49 511 47 124 712“),... }

Relationale Algebra

Operationsarten:

- Datenverwaltung (Datendefinition, Datenmanipulation, Zugriffs-, Integritäts- und Transaktionskontrolle)
- Anfragen (engl. queries)

Relationenalgebra prozedurale Sprache, in der spezifiziert wird, <u>wie eine Anfrage auszuwerten ist.</u>

Selektion

Es werden Zeilen einer Relation ausgewählt, für die **eine Bedingung** zutrifft.

Im Beispiel: - Ort = 'Hannover'

Notation: - $\sigma_{Ort='Hannover'}$(Kunde)

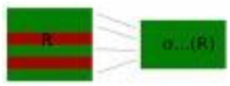
Ergebnis ist eine neue **Relation** mit **gleichem Schema**

Kunde	Kdnr	Name	Ort
	1	Michael Müller	Hannover
	2	Ulla Unken	Hamburg
	3	Karl Kraus	Berlin
	4	Lisa Lustig	Hannover

⬇

Resultat	Kdnr	Name	Ort
	1	Michael Müller	Hannover
	4	Lisa Lustig	Hannover

- Gesucht sind alle Einträge mit bestimmten Eigenschaften
- Auswahl von Zeilen (Tupel) einer Relation über Prädikate.
  Schreibweise: $\sigma_P(R)$
- Prädikate sind **boolesche Ausdrücke**
- Prädikate bestehen aus:
  - Operanden (Konstanten oder Attributnamen)
  - Vergleichs-Operatoren (<, =, >, ≤, ≠, ≥)
  - Logische Operatoren (¬, ∧, ∨)
- **Definition:** $\sigma_P(R) = \{ t \mid t \in R \land P(t) \}$



Projektion

Es werden **bestimmte Spalten** einer Relation ausgewählt.

Im Beispiel: - Name

Notation: - π_{Name}(Kunde)

Ergebnis ist eine neue **Relation** mit **anderem Schema**

Kunde	Kdnr	Name	Ort
	1	Michael Müller	Hannover
	2	Ulla Unken	Hamburg
	3	Karl Kraus	Berlin
	4	Lisa Lustig	Hannover

⬇

Resultat	Name
	Michael Müller
	Ulla Unken
	Karl Kraus
	Lisa Lustig

- Es interessieren oft nicht alle Spalten einer Relation
- Auswahl der Spalten (Attribute) $A_1, A_2, \ldots, A_k$ aus einer Relation R mit Grad n ≥ k.
  $\pi_{A1,\, A2,\, \ldots,\, Ak}(R)$
- **Definition:** $\pi_{A1,\, A2,\, \ldots,\, Ak}(R) = \{ p \mid \exists\, t \in R \text{ mit } p=(t(A1), \ldots, t(Ak)) \}$

Bsp. $\sigma_{Gehalt > 50000}(\pi_{Name,\, Gehalt,\, Alter}(Person))$

Kreuzprodukt

Kartesisches Produkt zweier Mengen: alle möglichen Paare, deren erstes Element aus der einen Menge und deren zweites Element aus der anderen Menge stammt

- Beispiel 1:
  - <u>M1 = {1,2,3} M2 ={a,b}</u>
  - M1 × M2 = { (1,a), (1,b), (2,a), (2,b), (3,a), (3,b) }

Wenn in einer Menge bereits Tupel sind, dann werden sie im Bereich Datenbanken i.d.R. „flachgeklopft“

- Beispiel 2:
  - <u>M1 = { (1,2), (3,4) } M2 ={ (a,x), (b,y) }</u>
  - M1 × M2 = { ((1,2), (a,x)), ((3,4), (a,x)), ((1,2), (b,y)), ((3,4), (b,y)) }
    = { (1,2,a,x), (3,4,a,x), (1,2,b,y), (3,4,b,y) }

Kombinieren von Kunden und Telefonnummern

Kunde	Kdnr	Name	Ort
	1	Michael Müller	Hannover
	2	Ulla Unken	Hamburg
	3	Karl Kraus	Berlin
	4	Lisa Lustig	Hannover

Telefon	Kdnr	Telefon
	1	12345678
	2	22334455
	3	13243546

⇒

Alle Kunden mit allen Telefonnummern

Resultat	Kdnr	Name	Ort	Kdnr2	Telefon
	1	Michael Müller	Hannover	1	12345678
	1	Michael Müller	Hannover	2	22334455
	1	Michael Müller	Hannover	3	13243546
	2	Ulla Unken	Hamburg	1	12345678
	2	Ulla Unken	Hamburg	2	22334455
	2	Ulla Unken	Hamburg	3	13243546
	3	Karl Kraus	Berlin	1	12345678
	3	Karl Kraus	Berlin	2	22334455
	3	Karl Kraus	Berlin	3	13243546
	4	Lisa Lustig	Hannover	1	12345678
	4	Lisa Lustig	Hannover	2	22334455
	4	Lisa Lustig	Hannover	3	13243546

Kombination mit Selektion und Projektion

- $\pi_{\text{Kunde.Kdnr, Name, Ort, Telefon}}(\sigma_{\text{Kunde.Kdnr = Telefon.Kdnr}}(\text{Kunde} \times \text{Telefon}))$

Resultat	Kdnr	Name	Ort	Telefon
	1	Michael Müller	Hannover	12345678
	2	Ulla Unken	Hamburg	22334455
	3	Karl Kraus	Berlin	13243546

Verbundoperation (engl. Join): $\text{Kunde} \bowtie_{\text{Kunde.Kdnr = Telefon.Kdnr}} \text{Telefon}$
Enspricht: $\sigma_{\text{Kunde.Kdnr = Telefon.Kdnr}}(\text{Kunde} \times \text{Telefon})$

Verbund

Kartesisches Produkt zwischen zwei Relationen R mit Grad r und Relation S mit Grad s, eingeschränkt durch eine Θ -Bedingung zwischen zwei Spalten A und B Vergleichsoperator $\Theta \in \{<, =, >, \leq, \neq, \geq\}$

- Θ -Verbund zwischen R und S:
  $R \bowtie_{A\, \Theta\, B} S$
- Für Θ = '=' spricht man auch vom Gleichverbund (**Equi-Join**):
  $R \bowtie_{A = B} S$
- Es sind auch kompliziertere Bedingungen möglich:
  $R \bowtie_{A > B \vee A < C} S$

Natürlicher Verbund (engl. Natural Join)

Es werden diejenigen Tupel aus den Relationen R und S kombiniert, für die die Werte der Attribute gleichen Namens übereinstimmen. Im Ergebnis sind diese Attribute nur einmal vorhanden.
Gegeben:

- $R(A_1, A_2, ..., A_m, B_1, B_2, ..., B_n)$
- $S(B_1, B_2, ..., B_n, C_1, C_2, ..., C_k)$

natürlicher Verbund von R und S ist nun:

- $\pi_{A1, A2, ..., Am, R.B1, ..., R.Bn, C1, ... Ck}(\sigma_{R.B1=S.B1 \wedge ... \wedge R.Bn=S.Bn}(R \times S))$

und wird folgendermaßen aufgeschrieben:

- **R⋈S**

Es kann $\underline{\pi_{\text{Kunde.Kdnr, Name, Ort, Telefon}}(\sigma_{\text{Kunde.Kdnr = Telefon.Kdnr}}(\text{Kunde} \times \text{Telefon}))}$ als **<u>Kunde ⋈ Telefon</u>** geschrieben werden.

Umbenennungen

Manchmal müssen Relationen oder Attribute umbenannt werden, damit Namenskonflikte aufgelöst werden
Qualifizierte Attributnamen bestehen aus Relationennamen und Attributnamen, getrennt durch einen Punkt:
Beispiele:

- StudentIn.Name
- ProfessorIn.Name

Voraus	Vorgänger	Nachfolger
	Java	OO-Program
	Java	JavaLabor
	OO-Program	Java-Projekt

Frage: Was ist die Vor-Voraussetzung für das Java-Projekt?

Umbenennung - Beispiel

Idee: kartesisches Produkt aus Voraus × Voraus und Herausfiltern des passenden Eintrags

	V_1		V_2	
Result	**Vorgänger**	**Nachfolger**	**Vorgänger**	**Nachfolger**
	Java	OO-Program	Java	OO-Program
	Java	JavaLabor	Java	OO-Program
	OO-Program	Java-Projekt	Java	OO-Program
	Java	OO-Program	Java	JavaLabor
	Java	JavaLabor	Java	JavaLabor
	OO-Program	Java-Projekt	Java	JavaLabor
	Java	OO-Program	OO-Program	Java-Projekt
	Java	JavaLabor	OO-Program	Java-Projekt
	OO-Program	Java-Projekt	OO-Program	Java-Projekt

Umbenennung – Definition und Schreibweise Umbenennungsoperator

$\rho_{\text{NeuerName}}(\text{AlterName})$

Beispiel

$\rho_{V1}(\text{Voraus})$ benennt die Tabelle Voraus in V1 um
$\rho_{V2}(\text{Voraus})$ benennt die Tabelle Voraus in V2 um

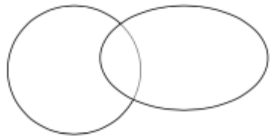
Ergebnis:

$\sigma_{\text{V2.Nachfolger="Java-Projekt"} \wedge \text{V1.Nachfolger=V2.Vorgänger}}(\rho_{V1}(\text{Voraus}) \times \rho_{V2}(\text{Voraus}))$

Mengenoperationen

Vereinigungsverträglichkeit der beteiligten Relationen muss gegeben sein:

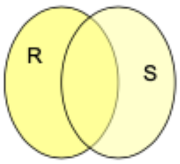
- Gleicher Grad der Relationen, d.h. dieselbe Anzahl Attribute
- Gleiche Bereiche, d.h. dieselben Domänen für die Attribute



<u>Typischen Mengenoperationen:</u>

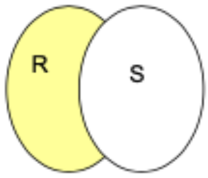
Vereinigung (UNION)

$R \cup S = \{t \mid t \in R \vee t \in S\}$



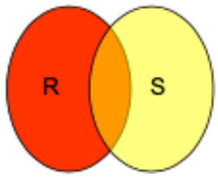
Differenz

$R - S = R \setminus S = \{t \mid t \in R \wedge t \notin S\}$



Durchschnitt

$R \cap S = R - (R - S) = \{t \mid t \in R \wedge t \in S\}$



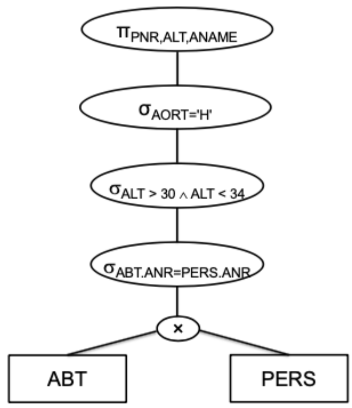
Anfrageoptimierung

Finde alle Angestellten (PNR, ALTER, ANAME), die in einer Abteilung in Hannover arbeiten und zwischen 30 und 34 Jahre alt sind.

$\pi_{PNR,ALT,ANAME}(\sigma_{AORT='H'}(\sigma_{ALT > 30 \wedge ALT < 34}(\sigma_{ABT.ANR=PERS.ANR}(ABT \times PERS))))$

$\pi_{PNR,\ ALT,\ ANAME}(\sigma_{ALT > 30 \wedge ALT < 34 \wedge AORT='H'}(ABT \bowtie_{ABT.ANR=PERS.ANR} PERS))$

$\pi_{PNR,\ ALT,\ ANAME}(\sigma_{AORT='H'}(ABT) \bowtie_{ABT.ANR=PERS.ANR\ \sigma ALT > 30 \wedge ALT < 34}(PERS))$



Operatorbaum

In einem Operatorbaum kann man sich veranschaulichen, wie und in welcher Reihenfolge die Operatoren ausgeführt werden.

Optimierungsregeln für Anfragen

- Führe Selektionen so früh wie möglich aus
- Führe Projektionen frühzeitig aus
- Fasse einfache Selektionen auf einer Relation zusammen
- Verknüpfe bestimmte Selektionen mit einem vorausgehenden kartesischen Produkt zu einem Verbund
- Bestimme die Verbundreihenfolge so, dass die Anzahl und die Größe der Zwischenobjekte minimiert wird

Äußerer Verbund

Bisher konnten Joins unvollständig sein, d.h. Tupel aus einer Relation waren im Join nicht vertreten, wenn sie keinen Partner gefunden haben.

Äußerer Verbund

• Nehme auch partnerlose Tupel mit auf

Äußerer Verbund: Versionen

- linker äußerer Verbund (left outer join):
  alle Tupel der linken Relation bleiben erhalten — $R ⟕_{Bedingung} S$
- rechter äußerer Verbund (right outer join):
  alle Tupel der rechten Relation bleiben erhalten — $R ⟖_{Bedingung} S$
- (vollständiger) äußerer Verbund (full outer join):
  alle Tupel bleiben erhalten — $R ⟗_{Bedingung} S$

Beispiel:

Outer Join - Beispiel

Abt	Anr	Aname	Aort
	K51	Planung	H
	K53	Einkauf	HI
	K55	Forschung	H

Abtl	Pnr	Name	Alter	Anr
	406	Coy	47	K60
	123	Müller	32	K51

Left Outer Join:

Result	Anr	Aname	Aort	Pnr	Name	Alter	Anr2
	K51	Planung	H	123	Müller	32	K51
	K53	Einkauf	HI				
	K55	Forschung	H				

Right Outer Join:

Result	Anr	Aname	Aort	Pnr	Name	Alter	Anr2
				406	Coy	47	K60
	K51	Planung	H	123	Müller	32	K51

Full Outer Join:

Result	Anr	Aname	Aort	Pnr	Name	Alter	Anr2
	K51	Planung	H	123	Müller	32	K51
	K53	Einkauf	HI				
	K55	Forschung	H				
				406	Coy	47	K60

Zusammenfassung Relationenalgebra

- Anfragen lassen sich in Relationenalgebra ausdrücken
  - Relationale Operatoren arbeiten auf Relationen (Tabellen)
  - Mit Hilfe der Projektion und der Selektion kann man Spalten und Zeilen einer Tabelle auswählen
  - Mehrere Tabellen können miteinander verknüpft werden:
    - Basisverknüpfung ist das kartesische Produkt
    - Ein Join filtert aus dem kartesischen Produkt die relevanten Einträge
  - Es gibt die aus der Mathematik bekannten Mengenoperationen
- Es gibt häufig verschiedene Ausdrücke für dieselbe Anfrage. Dann sollte man einen möglichst günstigen Ausdruck wählen.
  - Der Optimizer der Datenbank übernimmt diese Aufgabe!

Name	Bezeichnung	Definition
Selektion	$\sigma_P(R)$	$\{ t \mid t \in R \land P(t) \}$
Projektion	$\pi_{A1, A2, \ldots Ak}(R)$	$\{ p \mid \exists\, t \in R \text{ mit } p = (t(A_1), \ldots, t(A_k)) \}$
Kart. Produkt	$R \times S$	$\{ t \mid \exists\, r \in R, s \in S \text{ mit } t = (r,s) \}$
Natürlicher Verbund	$R \bowtie S$	$\pi_{A1, \ldots, Am, R.B1, \ldots, R.Bn, C1, \ldots Ck}(\sigma_{R.B1=S.B1 \land \ldots \land R.Bn=S.Bn}(R \times S))$
Theta-Verbund	$R \bowtie_{A \Theta B} S$	$\sigma_{A \Theta B}(R \times S)$
Equi-Verbund	$R \bowtie_{A = B} S$	$\sigma_{A = B}(R \times S)$
Umbenennung	$\rho_{Neu}(Alt)$	$\{ t \mid t \in Alt \}$
Vereinigung	$R \cup S$	$\{ t \mid t \in R \lor t \in S \}$
Durchschnitt	$R \cap S$	$\{ t \mid t \in R \land t \in S \}$
Differenz	$R - S$ oder $R \setminus S$	$\{ t \mid t \in R \land t \notin S \}$

Left Outer Join: $R ⟕_{\text{Bedingung}} S$
• Alle R-Tupel plus ggf. passende von S

Right Outer Join: $R ⟖_{\text{Bedingung}} S$
• Alle S-Tupel plus ggf. passende von R

Full Outer Join: $R ⟗_{\text{Bedingung}} S$
• Alle von R und S

Relationenmodell:
- Relationen mit Attributen (Tabellen)
- jedes Attribut hat einen Datentyp

Relationale Algebra:
- Operationen, um Daten aus den Tabellen abzufragen.
- Zentrale Operationen:
  - **Selektion, Projektion**
  - Kreuzprodukt -> **Verbund**
  - Zentrale Operationen:
- Schlüssel zur Identifikation von Tupeln
  Primärschlüssel
- Integritätsbedingungen
  Fremdschlüssel

Schlüssel

Insgesamt: „Superschlüssel => Schlüssel => Primärschlüssel“

Superschlüssel sind Teilmengen von Attributen, die ein Objekt eindeutig identifizieren

Ein **Schlüssel** ist ein Superschlüssel, der nicht verkleinert werden kann.

Unter allen Schlüsseln wird ein Schlüssel ausgewählt, der zur Identifikation verwendet wird. Dieser wird **Primärschlüssel** genannt. - Ein Schlüssel wird als Primärschlüssel ausgezeichnet.

Ein Primärschlüssel (primary key, auch PK abgekürzt) muss zwei wichtige Eigenschaften erfüllen:
- **Dauerhaft eindeutig**: Es gibt jede Ausprägung nur einmal (Schlüsseleigenschaft), auch in Zukunft!
- **Unveränderlich**: Die Attribute des PK einer Zeile ändern sich nicht

Integritätsbedingungen

Integritätsbedingungen sind Bestimmungen, die eingehalten werden müssen, um die Korrektheit und die logische Richtigkeit der Daten zu sichern.

Wahrheitsanforderungen:
- Können nur durch Vergleich mit der Realität überprüft werden

Beispiel: - Wohnt Albert Einstein wirklich in der Raumstr. 2, wie wir in der Tabelle Person gespeichert haben?

Logische Integritätsanforderungen:
- Betreffen die Gestalt der einzelnen Tabellen bzw. Relationen und die Beziehungen zwischen den verschiedenen Relationen

Beispiel: - Gibt es in der Tabelle Kunde überhaupt den Kunden mit der Id 16?

Logische Integritätsanforderungen

Lokale Integritätsbedingungen
- Bedingungen innerhalb **einer Relation**
- Es gibt nicht zwei identische Tupel in einer Tabelle:
  - Eine Relation ist eine Menge.
- Jeder Wert eines Attributs gehört zu einem definierten Wertebereich:
  - Für jedes Attribut wird ein Datentyp festgelegt
- Jedes Tupel muss eindeutig identifizierbar sein:
  - Festlegung eines Primärschlüssels

Globale Integritätsbedingungen
- Bedingungen, die über den Bereich einer **Relation hinaus reichen**
- Ein referenziertes Tupel einer anderen Relation muss existieren:
  - Referentielle Integrität

Referentielle Integrität

Referentielle Integritätsbedingung
- Ein Tupel einer Relation, auf das sich ein Tupel einer anderen Relation bezieht, muss vorhanden sein.

Beispiel: In die Tabelle ANG_PRO dürfen nur Personalnummern von Angestellten eingetragen werden, die auch in der Firma arbeiten

Ein **Fremdschlüssel** ist eine Attributmenge einer Relation, die in einer anderen Relation (Primär-)Schlüssel ist.

- Alle Attributwerte eines Fremdschlüssels tauchen in einer anderen Relation als Werte des (Primär-)Schlüssels auf.

- Telefon.Kdnr ist ein Fremdschlüssel auf Kunde.Kdnr
- Es dürfen also in Telefon.Kdnr nur Werte eingetragen werden, die auch in Kunde vorkommen

Kunde	Kdnr	Name	Ort
	1	Michael Müller	Hannover
	2	Ulla Unken	Hamburg
	3	Karl Kraus	Berlin
	4	Lisa Lustig	Hannover

referenzierter Primärschlüssel

Telefon	Kdnr	Telefon
	1	12345678
	2	22334455
	3	13243546

Fremdschlüssel