

**Definition: Prädikatenlogische Sprache erster Stufe (PL1- Sprache)**

(ohne Funktionssymbole)  
gegeben durch Paar  $(A, F)$  mit

- $A$  = Alphabet mit
  - unendlich viele Variable (aus einer Variablenmenge, genannt  $Var_A$ )
  - keine oder beliebig viele Konstantensymbole (aus einer Konstantenmenge  $Konst$ , genannt  $Konst_A$ )
  - mindestens ein Prädikatensymbol, möglicherweise unendlich viele Prädikatensymbole (aus einer Prädikatensymbolmenge, genannt  $Präd_A$ )
- $F$  = Menge aller wohlgeformten Formeln (Ausdrücke) mit Symbolen aus  $A$

**Definition: Term**

Ein Konstantensymbol aus  $Konst_A$  oder eine Variable aus  $Var_A$ .

**Definition: Atomare Formel (Atom)**

$p(t_1, \dots, t_n)$  mit  $p$  ist  $n$ -stelliges Prädikatensymbol aus  $A$  und  $t_1, \dots, t_n$  Terme

**Satz: Konstruktionsregel für F**

kleinste Menge mit

- Jedes Atom aus  $A$  ist in  $F$
- Falls  $f$  und  $g$  in  $F$  sind, dann sind auch  $\neg f, (f \wedge g), (f \vee g), (f \Rightarrow g), (f \Leftrightarrow g)$  in  $F$
- Falls  $X$  eine Variable ist und  $f$  in  $F$ , dann sind auch  $(\forall X)(f)$  und  $(\exists X)(f)$  in  $F$

## Definition: Interpretation

Eine Interpretation  $I$  für eine PL1-Sprache  $(A, F)$  ist ein Tripel  $(Dom, k, ext)$  mit:

- $dom$  ist eine nicht leere Menge, genannt Wertebereich (domain) von  $I$
- $k$  ist eine Abbildung von Konstantensymbolen aus  $A$  in  $dom$
- $ext$  (extension) ist eine Abbildung von Prädikatsymbolen aus  $A$  in Mengen von Tupeln, gebildet aus Werten von  $Dom$  unter Beachtung der Stelligkeit der Prädikatsymbole. Diese Tupelmengen heißen **Prädikate**,  $ext(p)$  wird auch **Ausprägung** von  $p$  genannt.

## Definition: Wahrheitswerte für PL1

**Konstanten und Variablen**  $\parallel t_1 \parallel_{I,\rho}: Var_A \cup Konst_A \rightarrow Dom$  definiert durch:

- $k(c)$  für Konstantensymbole
- $p(X)$  für Variablen

**Relationen**  $\models_{I,\rho} p(t_1, \dots, t_n)$  (Unter der Interpretation  $I$  und Belegung  $\rho$  wahr) genau dann wenn  $(\parallel t_1 \parallel_{I,\rho}, \dots, \parallel t_n \parallel_{I,\rho}) \in ext(\rho)$  für jedes Atom  $p(t_1, \dots, t_n)$  aus  $F$

## Definition: Folgerung ( $F' \models f$ )

Formel  $f \in F$  mit  $F' \subseteq F$ , falls jedes Modell von  $F'$  auch ein Modell von  $f$  ist.

## Definition: unter Folgerbarkeit abgeschlossen

Transitive Hülle von Folgerung

## Definition: Theorie

Besteht aus PL1-Sprache  $(A, F)$  und Formelmenge  $F' \subseteq F$ , die unter Folgerbarkeit abgeschlossen ist. ( $F'$ : Sätze der Theorie)

## Definition: Axiomatisierbar

Theorie  $T$ , falls es eine entscheidbare Formelmenge  $X \subseteq T$  gibt, derart dass alle Sätze von  $T$  Folgerungen von  $X$  (dann Axiomensystem von  $T$  mit Elementen Axiome) sind

## Satz: Modell einer Theorie

Eine Interpretation  $I$  ist ein Modell einer Theorie  $T$ , falls  $I$  ein Modell der Menge aller Sätze von  $T$  ist.

## Definition: Konsistente Theorie

Falls Theorie wenigstens ein Modell hat, sonst inkonsistent.

## Definition: formale Ableitung

Eine formale Ableitung einer Formel  $f$  von der Menge  $\{f_1, \dots, f_m\}$  in der PL1- Logik ist eine Folge  $b_1, \dots, b_l$  von Formeln der Sprache  $(A, F)$  mit

- $b_l$  ist gleich  $f$
- jede Formel  $b_j, j \in \{1, \dots, l\}$  ist eine der Formeln  $f_i, i \in \{1, \dots, m\}$  oder eine allgemeingültige Formel, die aus  $b_j$  in der Liste vorangehender Formeln durch Anwendung einer Ableitungsregel der PL1-Logik erhalten werden kann.

## Definition: ableitbar ( $\vdash$ )

Falls es eine formale Ableitung gibt (aus  $\emptyset$  oder Formelmenge)

## Satz: Vollständigkeitssatz von Gödel

$$\emptyset \models f \implies \vdash f$$

## Definition: Relationale Sprache

$R = (A, F)$  mit

- Es gibt in  $A$ 
  - $1 \leq |Konst_A| < \infty$
  - $|Präd_A| < \infty$
  - ausgezeichnetes, zweistelliges Prädikatsymbol = (Gleichheit)
  - ausgezeichnete Teilmenge einstelliger Prädikatsymbole, die **einfachen Typen**
- Typen, kleinste Menge mit
  - jeder einfache Typ von  $A$  ist ein Typ von  $R$
  - falls  $\tau_1, \tau_2$  Typen von  $R$ , dann auch  $(\tau_1 \wedge \tau_2), (\tau_1 \vee \tau_2), \neg \tau_1$  Typen von  $R$

## Definition: Relationale Interpretation

Sei  $R = (A, F)$  relationale Sprache, dann ist eine Interpretation  $I = (Dom, k, ext)$  für  $R$  eine relationale Interpretation für  $R$ , wenn

- $k$  ist eine Bijektion ( $Dom$  ist endlich)
- $ext(=) = \{(d, d) | d \in Dom\}$

## Definition: Relationale Datenbank

Tripel  $(R, I, IB)$  mit

- $R$  ist eine relationale Sprache
- $I$  ist eine relationale Interpretation
- $IB$  ist eine Menge von Formeln von  $R$ , so dass insbesondere für jedes  $n$ -stellige Prädikatsymbol  $P$ , das verschieden ist von " $=$ " und von den einfachen Typen,  $IB$  eine Formel der folgenden Gestalt enthalten muss ( $\tau_1, \dots, \tau_n$  einfache Typen):

$$(\forall X_1) \cdots (\forall X_n) (p(X_1, \dots, X_n) \Rightarrow \tau_1(X_1) \wedge \cdots \wedge \tau_n(X_n))$$

## Definition: Erlaubte relationale Datenbank

Falls I ein Modell von IB ist

## Definition: Intentionale, extentionale Prädikatsymbole

- Intentional: durch ein Programm definiert
- Extentional: als Relationen in einer Datenbank gespeichert

## Definition: Fixpunkttheorem (Knaster / Tarski)

Sei  $\tau$  eine monotone Transformation auf einem vollständigen Verband  $(V, \leq)$ . Dann hat  $\tau$  einen kleinsten Fixpunkt

$$lfp(\tau) = \inf(\{x \in V \mid \tau(x) \leq x\})$$

## Definition: Datalog Programm

Ein Datalog-Programm P (ohne IBen(Integritätsbedingungen)) ist eine endliche Menge von Horn-Klauseln mit Jedes  $d \in P$  ist entweder

- ein Fakt  $q(\dots)$ . ohne Variable
- eine sichere Regel  $q(\dots) : \neg p_1(\dots), \dots, p_n(\dots)$ . mit  $q \in iPraedikat$

Eine Regel heißt sicher, wenn alle in ihr vorkommenden Variablen beschränkt sind.

## Definition: Bedeutung eines Datalog Programms

Menge der Grundatome, die logisch aus P gefolgert werden können.

## Satz von Gödel / Skolem

Eine Klauselmenge P hat ein Modell genau dann wenn P hat ein Herbrand-Modell. Daraus folgt, dass ein Verfahren analog zu Wahrheitstabellen in der Aussagenlogik möglich ist.

## Skolemisierung

Jeder Formel der PL1 Logik, kann in eine erfüllbarkeitsäquivalente Formel in Skolem-Form gebracht werden. Dies bedeutet Pränexnormalform und alle Existenzquantoren durch Funktionen ersetzen.

## Definition: Herbrand-Interpretation

Eine Teilmenge der Herbrand- Basis

## Grundatom

Ein Grundatom  $f$  ist eine logische Folgerung einer Menge  $D$  von Datalog Klauseln (z.B.  $D \models f$ )  $\diamond_{Def}$ . Jedes Herbrand Modell von  $D$  ist auch ein Modell von  $f$ .

Da  $f$  ein Grundatom ist gilt  $D \models f \implies f$  ist in jedem Herbrand-Modell von  $D$  enthalten. Das heißt  $f \in \bigcap \{I \mid I \text{ Herbrand-Modell von } D\}$ .

Sei  $f \in \bigcap \{I \mid I \text{ Herbrand-Modell von } D\}$ , dann ist  $f$  ein Grundatom und jedes Modell von  $D$  auch in Modell von  $f$ .

## Definition: Menge aller Konsequenzen

$$cons(D) =_{def} \{f \in HB_D \mid D \models f\}$$

## Definition: Substitution

Eine Substitution ist eine endliche Menge der Form

$$\{X_1/t_1, \dots, X_n/t_n\}, X_1, \dots, X_n \text{ unterschiedliche Variablen, } t_1, \dots, t_n \text{ Terme, } X_i \neq t_i \quad (1)$$

Sei  $\theta$  eine Substitution,  $t$  ein Term (Variable oder Konstante), so gilt

$$t\theta =_{def} \begin{cases} t_i, & \text{falls } t/t_i \in \theta \\ t, & \text{sonst} \end{cases} \quad (2)$$

## Definition: Grundsubstitution

Substitution bei der alle  $t_i$  Konstanten sind.

## Definition: Unifizierbar

Seien  $L_1$  und  $L_2$  heißen **unifizierbar**, wenn  $(\exists \text{ Substitution } \Theta)(L_1\Theta = L_2\Theta)$ .  $\Theta$  heißt dann **Unifikator**.

## Definition: Komposition

Sei  $\Theta = \{X_1/t_1, \dots, X_n/t_n\}, \sigma = \{Y_1/n_1, \dots, Y_m/t_m\}$  Substitutionen.  
Die Komposition  $\Theta\sigma$  von  $\Theta$  und  $\sigma$  erhält man aus

$$X_1/t_1\sigma, \dots, X_m/t_m\sigma, Y_1/n_q, \dots, Y_m/n_m \quad (3)$$

Durch Streichen von Elementen der Form  $Z/Z$  sowie  $Y_i/n_i$  mit  $Y_i = X_j$  für ein  $j \in \{1, \dots, n\}$

## Definition: allgemeinere Substitution

Seien  $\Theta, \varsigma$  Substitutionen,  $\Theta$  heißt allgemeiner als  $\varsigma$   $(\exists \text{ Substitution } \delta)(\Theta\delta = \varsigma)$ .  
Seine  $L_1, L_2$  Literale. Ein allgemeinsten Unifikator (mgu) von  $L_1$  in  $L_2$  ist ein Unifikator von  $L_1$  und  $L_2$ , der allgemeiner als alle anderen Unifikatoren ist.

## Definition: Beweisbaum

B entsteht aus S durch Anwendung von  $\Theta$  auf alle Benennungen von Zielknoten.  
B repräsentiert einen Beweis für  $g\Theta$ , g benennung der Wurzel von S.

## Definition: Tiefe eines Baums

maximale Anzahl von Zielknoten auf einem Pfad von einem Blattknoten zur Wurzel.  
Entsprechend Knoten der Tiefe i, Ebene i eines Baumes. Zusätzlich: Spezielle Suchbäume (Tiefe 0) für Fakten aus P.

## Suchbaum zu cons

Sei  $P$  ein Datalog-Programm. Die Suchbaum / Beweisbaum Methode, angewandt auf alle Ziele  $q(X_1, \dots, X_{Stelligkeit(q)})$ ,  $q$  intentionales Prädikatesymbol von  $P$ , liefert  $cons(P)$  als Ergebnis

## Suchbaum, Vollständigkeit

Die Suchbaum / Beweisbaum Methode bleibt vollständig für ein Programm  $P$ , wenn nur Bäume mit max. Tiefe  $max\_fakt(P)$  betrachtet werden.

## Resolutionsmethode

Für allgemeine Klauselformen entwickelte Methode zum automatischen Beweisen.

## Definition: Vollständiger Verband

Partiell geordnete Menge  $(V, \leq)$  bei der zu jeder Teilmenge ein Infimum ( $\perp_V$ ) & Supremum ( $\top_V$ ) besteht. Jeder endliche Verband (und jeder Teilmengenverband) ist vollständig.

## Definition: Monotone Transformation

Abbildung  $\tau$  mit  $(\forall a, b \in V)(a \leq b \Rightarrow \tau(a) \leq \tau(b))$ .

## Definition: Fixpunkt

$a \in V : \tau(a) = a$



## Satz: Fixpunktttheorem (Knaster / Tarski)

Sei  $\tau$  eine monotone Transformation auf einem vollständigen Verband  $(V, \leq)$ . Dann hat  $\tau$  einen kleinsten Fixpunkt

$$lfp(\tau) = \inf(\{x \in V \mid \tau(x) \leq x\})$$

## Magic Set Methode

Transformiere ein Programm in eine Version, die für ein gegebenes Ziel die gleiche Ausgabe hat aber das Ziel bei bottom-up Auswertung berücksichtigt wird. Algorithmus, Beispiel hier.

### Vorgehen

**1.Schritt** Füge für das Ziel  $g = q(\dots)$  die Regel  $query^{f \dots 1}(X_1, \dots, X_k) : -q^a(\dots)$  ein, wobei  $X_1, \dots, X_k$  Variablen aus  $q^a(\dots)$  sind. Erzeuge für jede Regel  $r \in P$  und jedes mögliche Bindungsmuster  $\beta$  des Prädikates im Kopf von  $r$  eine Regel mit Bindungsmuster für jedes ihrer intensionalen Prädikate. Bestimme dabei unter Beachtung von  $\beta$  für jedes Argument im Rumpf ob es ausgezeichnet ist oder nicht. Falls ein IDB-Prädikat im Rumpf mehrfach auftritt, sollte man es durchnummerieren.

**2.Schritt** Forme  $P_g^{B2}$  zu  $P_g^{magic3}$ .

Sei  $P_g^{magic} := P_g^B$ . Mach dann für jedes  $r \in P_g^B$  und draus folgend für jedes Vorkommen  $p^\beta_i(t_1, \dots, t_l)$  eines IDB-Prädikates im Rumpf von  $r$  folgendes:

- Streiche alle anderen Vorkommen von IDB-Prädikaten im Rumpf von  $r$
- Ersetze  $p^\beta_i$  durch  $magic\_r\_p^\beta\_i$
- Streiche alle Variablen aus  $(t_1, \dots, t_l)$ , die nicht ausgezeichnet sind. <sup>4</sup>
- Streiche alle nicht ausgezeichneten EDB-Prädikate aus  $r$ .

<sup>1</sup>Hochgestellte Zeichen sind Bindungsmuster (wie in Coral)

<sup>2</sup>Menge aller erreichbaren Regeln aus Schritt 1

<sup>3</sup>Bezüglich  $g$  äquivalent

<sup>4</sup>Bei "Prädikaten" ohne Argumente die entstehen können: Fall entsprechende Relation  $\neq \emptyset$  wahr, sonst falsch

- Sei  $z^\alpha(s_1, \dots, s_k)$  das Prädikat im Kopf von  $r$ . Streiche alle Variablen aus  $(s_1, \dots, s_k)$ , die nicht ausgezeichnet sind;  $\alpha$  wird nicht verändert.  
Ersetze  $magic\_r\_p^\beta\_i(t_1, \dots, t_l)$  durch  $magic\_z^\alpha(s'_1, \dots, s'_l)$ <sup>5</sup>
- Füge  $P_g^{magic}$  die neuen Regeln hinzu

**3. + 4.Schritt**

---

<sup>5</sup>Änderungen aus letztem Schritt

```

for each  $r \in P_g^\beta$  do
  begin
    for each  $p^\beta \_i(t_1, \dots, t_l), p \in iPräd$  im Rumpf von  $r$  do
      begin erzeuge Prädikat  $m = magic\_r \_p^\beta \_i(t'_1, \dots, t'_l)$ ,
        wobei die  $t'_1, \dots, t'_l$  die ausgezeichneten Argumente von
           $t_1, \dots, t_l$  sind;
      if  $p$  Prädikatsymbol im Kopf von  $r$ 
      then füge  $m$  am Beginn des Rumpfes von  $r$  ein
      else füge  $m$  unmittelbar vor  $p^\beta \_i(t_1, \dots, t_l)$  ein
      end; /* Einfügeposition für Semantik ohne Bedeutung */
    ersetze Rumpf von  $r$  in  $P_g^{magic}$  durch den geänderten Rumpf
  end;

```

Figure 1:

```

for each  $r \in P_g^{magic}$  do
  for each  $p^\beta \_i(t_1, \dots, t_l)$  im Rumpf von  $r$  do
     $P_g^{magic} := P_g^{magic} \cup \{magic\_p^\beta(t'_1, \dots, t'_l) : - \quad magic\_r \_p^\beta \_i(t'_1, \dots, t'_l)\}$ ;

```

Figure 2:

## Definition: Ausgezeichnet

**Argument eines Teilziels** Konstantensymbol, gemäß  $\alpha$  gebunden, es in einem EDB-Prädikat auftritt, das ein ausgezeichnetes Argument hat.

**EDB-Prädikat** Alle seine Argumente sind ausgezeichnet

## Definition: Abhängigkeitsgraph

Gerichteter Graph  $DG(P) = (V, E)$  eines Programms  $P$ , falls  $V$  die Menge aller Prädikatsymbole von  $P$  und  $e = (p, q) \in E \Leftrightarrow q$  kommt im Rumpf einer Regel von  $P$  vor, deren Kopfprädikat  $p$  ist.  $e$  ist mit “ $-$ ” benannt, wenn  $q$  dabei wenigstens einmal negiert vorkommt.

## Definition: Schichtung eines Programms

Eine Folge  $\Pi_1, \dots, \Pi_n$  von Mengen von Prädikatsymbolen mit

- $\{\Pi_1, \dots, \Pi_n\}$  ist eine Partition der Prädikatsymbole in  $P$
- $q \in \Pi_i, r \in \Pi_j, (q, r)$  Kante in  $DG(P) \Rightarrow i \geq j$ , für alle Prädikatsymbole  $q, r$  aus  $P, i, j \in \{1, \dots, n\}$
- $q \in \Pi_i, r \in \Pi_j, (q, r)$  mit “ $\neg$ ” markierte Kante in  $DG(P) \Rightarrow i > j$

## Definition: Geschichtetes Programm

Ein Programm  $P$  heißt geschichtet, wenn  $P$  eine Schichtung hat.

## Satz: Schichtung / Zyklus

Ein Programm  $P$  ist geschichtet, gdw.  $DG(P)$  enthält keinen Zyklus mit einer Kante, die mit “ $\neg$ ” markiert ist.

## Eigenschaften: Perfektes Modell

“Kleinstes” der minimalen Modelle, wenn das stärkste Gewicht darauf gelegt wird, dass die Prädikate niedriger Schichten klein bleiben

## Satz: sicher geschichtet / perfektes Modell

Sei  $P$  ein sicheres, geschichtetes Programm, dann ist das Perfekte Modell von  $P$  unabhängig von der Wahl der Schichtungen

## Definition: Anfrage

Sei  $\sigma = \{(RT_1, \alpha_1), \dots, (RT_n, \alpha_n)\}$  ein relationales Datenbankschema über  $\alpha = \bigcup_{i=1}^n \alpha_i$  mit Wertebereichsfunktion  $\text{dom}$ . Sei  $\alpha$  in eine genügend große Attribut-

menge  $\alpha_0$  eingebettet<sup>6</sup>.

Eine Anfrage  $q$  auf  $\sigma$  ist eine partielle Funktion

$$q|Z\sigma \Rightarrow R_\beta^\infty$$

## Definition: Anfragesprache

Eine Anfragesprache zu  $\sigma$  ist eine Menge  $L_0$  von Ausdrücken zusammen mit einer Bedeutungsfunktion (in Zeichen:  $(L_\sigma^7, \mu^8)$ ), so dass für jeden Ausdruck  $e \in L_\sigma$  gilt:  $\mu(e)$  ist eine Anfrage von  $\sigma$

## Definition: Ausdruckskraft

Die Ausdruckskraft einer Anfragesprache  $(L_\sigma, \mu)$  zu einer DB-Schema  $\sigma$  ist definiert als  $\mu(L_\sigma) =_{Def.} \{\mu(e) | e \in L_\sigma\}$ . Eine Sprache  $(L_\sigma, \mu')$  ist **ausdrucksstärker** als eine Sprache  $(L_\sigma, \mu)$ , wenn gilt:  $\mu(L_\sigma) \subseteq \mu'(L'_\sigma)$  Im Fall  $\mu(L_\sigma) = \mu'(L'_\sigma)$  werden die Sprachen **äquivalent** genannt

## Definition: Anfragesprache von $\rho$

Sei  $\rho = \{(RT_i, \alpha_i) | i \in Lm\}$  ein relationales Datenbankschema über  $\alpha = \bigcup \alpha_i$  mit Werteberichsfunktion dom. Sei  $\alpha$  in eine genügend große Attributmenge  $\alpha_0$  eingebettet. Eine Anfrage  $q$  ist eine partielle Funktion mit  $q : \delta_\rho \rightarrow R_\beta^\infty, \delta \subseteq \alpha_0, R_\beta^\infty$  ist die Menge der verallgemeinerten DB-Relationen über  $\beta$  (unendliche Teilmengen sind erlaubt).

Eine Anfragesprache zu  $\rho$  ist eine Menge  $L_\rho$  von Ausdrücken mit einer Bedeutungsfunktion  $\mu$  (schreibe  $(L_\rho, \mu)$ ), so dass für jeden Ausdruck  $e \in L_\rho$  gilt  $\mu(e)$  ist eine Anfrage an  $\rho$ .

---

<sup>6</sup>Enthalten darin

<sup>7</sup>Sprache

<sup>8</sup>Bedeutungsfunktion

## Definition: Ausdruckskraft einer Anfragesprache von $\rho$

Die Ausdruckskraft einer Anfragesprache  $(L_\rho, \mu)$  zu  $\rho$  ist definiert als  $\mu(L_\rho) = \{\mu(e) \mid e \in L_\rho\}$

## Definition: Äquivalenz von Sprachen

$(L'_\rho, \mu') = (L_\rho, \mu)$ , wenn  $\mu'(L'_\rho) = \mu(L_\rho)$ . Kleiner und größer analog. Falls Anfragesprache  $L$  für alle  $\rho$  äquivalent zu Anfragesprache  $L'$  ist werden beide als äquivalent bezeichnet.

## Satz 2.1

Sei  $e$  ein zu einem gegebenen Datenbankschema  $\rho$  passender RA-Ausdruck ohne Komplement. Dann gibt es einen äquivalenten zu  $e$  passenden Ausdruck der RA  $e'$  in dem als Operatoren nur Selektionen mit einfachen Vergleichsausdrücken, direktes Produkt, Projektion, Umbenennung, Differenz, Vereinigung vorkommen. Als Operanden ermittelt  $e'$  neben Relationstypbezeichnern aus  $\rho$  nur (extensionale) DB-Relationen der Form  $\{(A, C), A \in \alpha_\rho, c \in \text{dom}(A)\}$

Weitere Einschränkungen sind möglich  $\rightarrow$  Projektionen nur auf ein Attribut, Vergleichsausdrücke  $=$  und  $<$ .

Falls Komplementbildung hinzugenommen, Differenz nicht mehr notwendig:

$$R \setminus S = (R[\text{kompl}] \cup S)[\text{kompl}]$$

## Satz 2.2

Eine derart ??? Operationsmenge der RA ist minimal, d.h. es kann keine Operation entfernt werden, ohne dass die Ausdruckskraft eingeschränkt ist.

## Satz 2.3

Sei  $RT$  der Bezeichner eines beliebigen, zweistelligen Relationstyp über abzählbaren unendlichen Wertebereich. Sei  $Rt^*$  der Relationstyp für die transitive Hülle von

Relationen des Typs RT. Es gibt keinen Ausdruck der Relationenalgebra mit der Eigenschaft  $e(RT)? = RT*$ .

Gegeben Wertebereich  $\{a_1, a_2, \dots\}$  ohne Ordnungsrelation. Betrachte  $R_l = \{(a_i, a_i + 1) | i \in [l - 1]\}$  für  $l \in \mathbb{N}$ .  $R_l$  ist eine mathematische Relation, die Tupel sind also geordnet. Zeige  $e(R_l) \neq R_l*$  für jeden RA - Ausdruck  $e$ , der zu Wertebereichen passt und für genügend großes  $l$ .  $e(R_l)$  bedeutet  $R_l$  für RT eingesetzt. RA-Operationen sind anzupassen (wir haben keine Bezeichner  $\leadsto$  Permutationen einführen) .

- Projektion: <sup>9</sup> erlaubte Permutationen (vgl  $p(x) :- r(y,x)$  ist Projektion in  $r$ )
- Selektionen mit atomaren Vergleichsausdrücken der Form  $i = a_m, i \neq a_m, i = j, i \neq j$  für  $i, j \in [k], m \in [l]$   $k$  ist bestimmt durch Größe der konstruierten Tupel  $(b_1, \dots, b_k), b_i \in \{a_1, \dots, a_l\}$

## Lemma 2.1

Sei  $e$  ein beliebiger RA-Ausdruck, der zum Wertebereich  $\{a_1, a_2, a_3, \dots\}$  passt und zu RT. Dann lässt sich  $e(R_l)$  für ein genügend großes  $l$  darstellen als

$$e(R_l) = \{(X_1, \dots, X_K) / \Psi(X_1, \dots, X_K)\} \subseteq \{a_1, \dots, a_l\}^K$$

mit  $K \in \mathbb{N}$ ,  $\Psi$  aussagenlogischer Ausdruck in disjunktiver Form, wobei atomare Ausdrücke nur Vergleichsausdrücke der oben genannten Form auftreten.

## Definition: Vollständigkeit einer Anfragesprache

Eine Anfragesprache  $(L_0, \mu)$  zu einem DB-Schema  $\sigma$  heißt vollständig, wenn gilt:

1. Für jeden Ausdruck  $e \in L_\sigma$  ist  $\mu(e)$  berechenbar und generisch
2. Für jede berechenbare und generische Anfrage  $q$  an  $\sigma$  gibt es einen Ausdruck  $e \in L_\sigma$  mit  $\mu(e) = q$ .

---

<sup>9</sup>entspricht  $\exists$ : es gibt da was in der spalte, aber was das ist, ist mir egal

## Definition: Abgeschlossenheit einer Anfragesprache

Eine allgemeine Anfragesprache  $L$  mit Interpretationsvorschrift  $\mu$  heißt abgeschlossen, wenn sie zu jedem Datenbank-Schema  $\sigma$  eine vollständige Anfragesprache  $(L_\sigma, \mu)$  enthält. Offensichtlich:

- Alle RA-Anfragen sind berechenbar und generisch
- Die RA ist nicht abgeschlossen (s. Satz 2.3)

## Definition: QL-Terme

Die Menge der **QL-Terme** ist induktiv definiert:

1.  $E$  (Gleichheit) ist ein Term,  $r_i, x_i$  sind Terme
2. Falls  $e, e'$  Terme sind, dann  $(e \cap e'), (\neg e), (e \downarrow), (e \uparrow)$  und  $e \circ$

## Definition: QL-Programme

Die Menge der **QL-Programme** ist induktiv definiert durch

1.  $x_i := e$  ist ein Programm für einen Term  $e$  und  $i \geq 1$
2. Für Programme  $P, P'$  sind  $(P; P')$  und **while**  $x_i$  **do**  $P$  Programme. Es gelten die üblichen Klammereinsparungsregeln

## Definition: QL-Semantik

Seien  $\sigma = \{(RT_1, s_1), \dots, (RT_m, s_m)\}$ ,  $s_i \in \mathbb{N}$ ,  $z \in \xi_\sigma$ .

- $R_s^\emptyset$  : leere Relation der Stelligkeit  $s$
- $R_s$  : Menge der Relationen mit Stelligkeit
- $R_{-1} = \{R_0^\emptyset\}$

$\leadsto E \in R_2$  mit  $E = \{(d, d) | d \in Dom\}$



- $r_i = \begin{cases} z(RT_i) & \text{falls } i \leq m \\ R_0^\emptyset, & \text{sonst} \end{cases}$
- “ $\cap$ ” entspricht Durchschnitt, falls beide Argumente gleiche Stelligkeit haben, sonst  $R_0^\emptyset$
- “ $\neg$ ”:  $R_S \rightarrow R_S: \neg e =_{def} Dom^s - e$
- “ $\downarrow$ ”  $R_S \rightarrow R_{s-1}: e \downarrow =_{def} \{(d_2, \dots, d_s) | (d_1, \dots, d_s) \in e\}$
- “ $\uparrow$ ”  $R_S \rightarrow R_{s+1}: e \uparrow =_{def} \{(d_2, \dots, d_s, d) | (d_1, \dots, d_s) \in e, d \in Dom\}$
- “ $\circ$ ”  $R_S \rightarrow R_S: e \circ =_{def} \{(d_1, \dots, d_{s-2}, d_s, d_{s-1}) | (d_1, \dots, d_s) \in e\}$

## Satz 2.4

QL ist abgeschlossen

## Inklusionssatz

$$\begin{aligned} LOGSPACE &\subseteq^? NLOGSPACE \subseteq^? P \subseteq^? NP \\ &\subseteq^? PSPACE \subseteq NPSPACE \subseteq EXPTIME \subseteq REK \end{aligned}$$

- $P \rightsquigarrow$  explizite Anfrage
- $PSPACE \rightsquigarrow$  while Anfrage

## Definition: Datenkomplexität einer Anfrage

Die Datenkomplexität<sup>10</sup> einer Anfrage  $q$  ist die Komplexität ihres Erkennungsproblems. Größe des Ergebnisses (der Antwort) ist nicht erfasst. Falls gewünscht —**Antwortkomplexität**:  $\{code_E(z) + code_e(q(z)) | \dots\}$  Komplexität der Konstruktion des Ergebnisses erfasst.

Unterschied nicht wesentlich bei Komplexitätsklassen, die bzgl. Polynomialfaktor unempfindlich sind (ab “ $P$ “ aufwärts)

---

<sup>10</sup>kurz: Komplexität

**Aber** Mit Antwortkomplexität keine Unterscheidung zwischen leichten und schwierigen Anfragen, die große Ergebnisse haben möglich

## Definition: Vollständiges Problem:

schweres Problem (Problem lösen entspricht Menge entscheiden).

## Definition: hartes Problem

Ein Problem heißt hart für eine Komplexitätsklasse  $K$  unter einem Reduktionstyp, falls jedes Problem in  $K$  mit einer Reduktion des Typs auf  $p$  reduziert werden kann. Falls  $p$  in  $K$ :  $p$  ist  $K$ -vollständig

## Definition: Vollständigkeit bezüglich einer Komplexitätsklasse $K$

Eine Anfragesprache  $L_\sigma$  zu einem DBSchema  $\sigma$  ist vollständig bezüglich einer Komplexitätsklasse, falls gilt: Sei  $Q_\sigma K$  die Menge aller Anfragen an  $\sigma$  mit Komplexität  $K$ :

1.  $\{\mu(e) | e \in L_\sigma\} \subseteq Q_\sigma K$
2.  $(\exists e \in L_\sigma)(\text{das Erkennungsproblem für } e \text{ ist vollständig bezüglich } K)$

$L$  heißt vollständig bezüglich  $K$ , falls  $L_\sigma$  für jedes  $\sigma$  vollständig ist bezüglich  $K$ .

## Satz 2.9

Kalkülanfragen (RA-Anfragen) sind in QLOGSPACE

## Satz 2.10

$QNLOGSPACE = DRC^{pos\_trans}$

## Gleichheitsprädikat-Axiom

Gleichheit ist Äquivalenzrelation

## Leibnitz-Prinzip

$$(\forall X_1) \cdots (\forall X_n) (\forall Y_1) \cdots (\forall Y_n) ((p(X_1, \dots, X_n) \wedge \bigwedge_{i=1}^n (X_i, Y_i)) \Rightarrow p(Y_1, \dots, Y_n))$$

## Abschlussaxiom für Wertebereiche

$$Konst_A = \{c_1, \dots, c_m\}, (\forall X) (\bigvee_{i=1}^n (X, c_i))$$

## Eindeutigkeitsaxiom für Namen

$\neg = (c, c')$  für jedes Paar verschiedener Werte  $(c, c')$  aus  $Konst_A$

## Vollständigkeitsaxiom

Für jedes Prädikat  $p$  anzugeben

- $E_p \subseteq \chi_0$  Menge aller Grundatome aus  $\chi_0$
- $K_p = \{(c_1, \dots, c_n) | p(c_1, \dots, c_n) \in E_p\} = \{(c_{i1}, \dots, c_{in}) | i \in [m]\}$   
 $(\forall X_1) \cdots (\forall X_n) (p(X_1, \dots, X_n) \Rightarrow \bigvee_{j \in [m]} (\bigwedge_{i \in [n]} (X_i, c_{ij}))$   
 $K_p = \emptyset \Rightarrow (\forall X_1) \cdots (\forall X_n) (\neg p(X_1, \dots, X_n))$   
 Entspricht: Alles in DB wahr, alles andere falsch

## Axiome für rel. Sprache

- Menge der Grundatome mit Prädikatsymbolen, nicht =
- Vollständigkeitsaxiom für alles außer =

- Abschlussaxiom für  $Konst_A$
- Eindeutigkeitsaxiom für  $Konst_A$
- Gleichheitsaxiom (Reflexiv, Symmetrisch, Transitiv)
- nichts außer dem, was angegeben ist

## Eigenschaften der GCWA

**Konsistenz:** Negative Infos, die auf Grund der gcwa hinzugefügt werden führen nicht zu Widersprüchen in der Theorie

**Stabilität:** verringert und vergrößert positive Infos nicht

**Reduktion:** für definite Programme entspricht gcwa der cwa

**Keine Monotonie** : im Gegensatz zu cwa