Teil III.

Prädikatenlogik

5. Grundlagen der Prädikatenlogik

In diesem Kapitel führen wir die *Prädikatenlogik*, abgekürzt FO für „first-order logic“, ein. Die Prädikatenlogik macht Aussagen über Strukturen und Elemente von Strukturen. Insbesondere also sind Formeln nicht einfach nur noch wahr oder falsch.

5.1. Syntax der Prädikatenlogik

Wir führen zunächst die Syntax der Prädikatenlogik ein. Dazu brauchen wir zunächst den Begriff der Variablen sowie der Terme.

Definition 5.1 (Variablen und Terme) Wir fixieren die Menge $\textsc{Var} := \{v_i : i \in \mathbb{N}\}$ von *Variablen erster Stufe*, kurz *Variablen*. wir mit VAR.

Sei σ eine Signatur. Die Menge $\mathcal{T}_\sigma$ der σ-Terme ist induktiv wie folgt definiert.

Basisfall

- $v_i \in \mathcal{T}_\sigma$ für alle $v_i \in \textsc{Var}$.
- $c \in \mathcal{T}_\sigma$ für alle Konstantensymbole $c \in \sigma$.

Induktionsschritt

- Ist $f \in \sigma$ ein k-stelliges Funktionssymbol und $t_1, \ldots, t_k \in \mathcal{T}_\sigma$ dann ist

$$f(t_1, \ldots, t_k) \in \mathcal{T}_\sigma\,.$$

Ein Term, in dem keine Variablen vorkommen, heißt *Grundterm*. ⊣

Definition 5.2 (Syntax der Prädikatenlogik) Sei σ eine Signatur. Die Menge $\textsc{FO}[\sigma]$ der *prädikatenlogischen Formeln über* σ ist induktiv wie folgt definiert.

Basisfall

- $t = t' \in \textsc{FO}[\sigma]$ für alle Terme $t, t' \in \mathcal{T}_\sigma$.

- $R(t_1, \ldots, t_k) \in \text{FO}[\sigma]$, für alle k-stelligen Relationssymbole $R \in \sigma$ und alle $t_1, \ldots, t_k \in \mathcal{T}_\sigma$.

Formeln der Form $t = t'$ und $R(t_1, \ldots, t_k)$ heißen *atomar*.

Induktionsschritt

- Wenn $\varphi \in \text{FO}[\sigma]$, dann $\neg\varphi \in \text{FO}[\sigma]$.
- Wenn $\varphi, \psi \in \text{FO}[\sigma]$, dann ist $(\varphi \vee \psi) \in \text{FO}[\sigma]$, $(\varphi \wedge \psi) \in \text{FO}[\sigma]$, $(\varphi \to \psi) \in \text{FO}[\sigma]$ und $(\varphi \leftrightarrow \psi) \in \text{FO}[\sigma]$.
- Wenn $\varphi \in \text{FO}[\sigma]$ und $x \in \text{VAR}$, dann ist $\exists x\varphi \in \text{FO}[\sigma]$ und $\forall x\varphi \in \text{FO}[\sigma]$.

$\text{FO}[\sigma]$ heißt die *Prädikatenlogik über σ* oder die *Sprache/Logik erster Stufe über σ*. ⊣

Wir betrachten einige Beispiele.

Beispiel 5.3 Sei $\sigma_{Graph} := \{E\}$, wobei E eine binäre Relation ist. Die folgenden Ausdrücke sind Formeln in $\text{FO}[\sigma_{Graph}]$, der Sprache der Graphen.

- $E(x, y)$
- $\exists x \forall y E(x, y)$
- $\exists x_1 \exists x_2 \big((E(x, x_1) \wedge E(x_1, x_2)) \wedge E(x_2, y) \big)$ ⊣

Beispiel 5.4 Sei $\sigma := \{<, +, *\}$, wobei $<$ eine binäre Relation und $+$ und $*$ binäre Funktionen sind. Die folgenden Ausdrücke sind Formeln in $\text{FO}[\sigma]$, der Sprache der Arithmetik mit Ordnung. Zur leichteren Lesbarkeit verwenden wir $<$ und $+$ in Infixnotation, auch wenn das streng genommen keine prädikatenlogischen Formeln sind.

- $x < x + x$
- $\forall x \exists y\ x < y$
- $\exists x \neg \exists y\ y < x$ ⊣

Definition 5.5 Sei σ eine Signatur. Wir schreiben $\text{var}(t)$ für die in einem σ-Term t vorkommenden Variablen. Formal wird $\text{var}(t)$ wie folgt induktiv definiert:

- Wenn $t = v_i \in \text{VAR}$ dann $\text{var}(t) := \{v_i\}$.
- Wenn $t = c$ für ein Konstantensymbol $c \in \sigma$ dann $\text{var}(t) := \varnothing$.
- Wenn $t = f(t_1, \ldots, t_k)$ für ein k-stelliges Funktionssymbol $f \in \sigma$ und Terme $t_1, \ldots, t_k \in \mathcal{T}_\sigma$ dann $\text{var}(t) := \text{var}(t_1) \cup \cdots \cup \text{var}(t_k)$. ⊣

Definition 5.6 (Freie und gebundene Variablen) Sei σ eine Signatur. Die Menge *frei*(φ) der *freien Variablen* einer Formel $\varphi \in \text{FO}[\sigma]$ ist induktiv definiert durch:

- Für $t_1, t_2 \in \mathcal{T}_\sigma$ ist $\mathit{frei}(t_1 = t_2) := \text{var}(t_1) \cup \text{var}(t_2)$.
- Wenn $\varphi = R(t_1, \ldots, t_k)$ für $R \in \sigma$ und $t_1, \ldots, t_k \in \mathcal{T}_\sigma$, dann $\mathit{frei}(\varphi) := \bigcup_{i=1}^{k} \text{var}(t_i)$.
- $\mathit{frei}(\neg\varphi) := \mathit{frei}(\varphi)$ für alle $\varphi \in \text{FO}[\sigma]$.
- $\mathit{frei}\big((\varphi * \psi)\big) := \mathit{frei}(\varphi) \cup \mathit{frei}(\psi)$ für alle $* \in \{\vee, \wedge, \to, \leftrightarrow\}$ und $\varphi, \psi \in \text{FO}[\sigma]$.
- Wenn $\varphi = \exists x\psi$ oder $\varphi = \forall x\psi$, für $x \in \text{VAR}$ and $\psi \in \text{FO}[\sigma]$, dann $\mathit{frei}(\varphi) := \mathit{frei}(\psi) \setminus \{x\}$.

Eine Formel φ mit $\mathit{frei}(\varphi) = \varnothing$ heißt ein *Satz*. Eine Variable, die in φ vorkommt, aber nicht frei ist, heißt *gebunden*. Wir schreiben $\varphi(v_1, \ldots, v_k)$, um zu sagen, dass $\mathit{frei}(\varphi) \subseteq \{v_1, \ldots, v_k\}$. ⊣

Beispiel 5.7 Betrachten wir noch einmal die Formeln aus den Beispielen 5.3 und 5.4.

- Für $\varphi = E(x, y)$ gilt $\mathit{frei}(\varphi) = \{x, y\}$.
- Für $\varphi = \exists x \forall y (x = y \vee E(x, y))$ gilt $\mathit{frei}(\varphi) = \varnothing$.
- Für $\varphi = \exists x_1 \exists x_2 \big((E(x, x_1) \wedge E(x_1, x_2)) \wedge E(x_2, y)\big)$ gilt $\mathit{frei}(\varphi) = \{x, y\}$.
- Für $\varphi = x < x + x$ gilt $\mathit{frei}(\varphi) = \{x\}$.
- Für $\varphi = \forall x \exists y\ x < y$ gilt $\mathit{frei}(\varphi) = \varnothing$.
- Für $\varphi = \exists x \neg \exists y\ y < x$ gilt $\mathit{frei}(\varphi) = \varnothing$. ⊣

5.2. Semantik der Prädikatenlogik

In der Aussagenlogik wurde die Semantik durch eine Belegung der Variablen mit Wahrheitswerten definiert. In der Prädikatenlogik werden wir ebenfalls Belegungen als Basis der Semantik verwenden, allerdings wird hier den Variablen ein Wert aus dem Universum der Struktur zugewiesen.

Definition 5.8 (Belegungen und Interpretationen) Sei σ eine Signatur und $\mathcal{A}$ eine σ-Struktur.

(1) Eine *Belegung* in $\mathcal{A}$ ist eine Funktion $\beta : \text{def}(\beta) \to A$ mit $\text{def}(\beta) \subseteq \text{VAR}$. β ist *passend* für $\varphi \in \text{FO}[\sigma]$, wenn $\mathit{frei}(\varphi) \subseteq \text{def}(\beta)$.

(2) Eine *σ-Interpretation* ist ein Paar $(\mathcal{A}, \beta)$, bestehend aus einer σ-Struktur $\mathcal{A}$ und einer Belegung β in $\mathcal{A}$. $\mathcal{I} := (\mathcal{A}, \beta)$ ist passend für $\varphi \in \text{FO}[\sigma]$, wenn β zu φ passt. ⊣

Wir vereinbaren als nächstes folgende Notation.

Notation 5.9 Sei $\mathcal{A}$ eine σ-Struktur.

(1) Ist β eine Belegung, $x \in \text{VAR}$ und $a \in A$ ein Element, dann definieren wir eine neue Belegung $\beta[x/a]$ mit $\text{def}(\beta[x/a]) := \text{def}(\beta) \cup \{x\}$ durch

$$\beta[x/a](y) := \begin{cases} a & \text{wenn } x = y \\ \beta(y) & \text{sonst.} \end{cases}$$

Die Belegung $\beta[x/a]$ ist also genauso wie β definiert, mit dem Unterschied, dass x nun durch a interpretiert wird.

(2) Ist $\mathcal{I} := (\mathcal{A}, \beta)$ eine σ-Interpretation, $x \in \text{VAR}$ und $a \in A$ ein Element, dann definieren wir $\mathcal{I}[x/a]$ als $(\mathcal{A}, \beta[x/a])$. ⊣

Definition 5.10 Sei σ eine Signatur. Induktiv über den Formelaufbau definieren wir eine Funktion $[\![\cdot]\!]^{\cdot}$, die jedem Term $t \in \mathcal{T}_\sigma$ und jeder σ-Interpretation $\mathcal{I} := (\mathcal{A}, \beta)$ für t einen Wert $[\![t]\!]^{\mathcal{I}} \in A$ zuweist.

Basisfall.

- $[\![v_i]\!]^{\mathcal{I}} := \beta(v_i)$ für alle $v_i \in \text{VAR}$
- $[\![c]\!]^{\mathcal{I}} := c^{\mathcal{A}}$ für alle Konstantensymbole $c \in \sigma$.

Induktionsschritt.

- Ist $f \in \sigma$ ein k-stelliges Funktionssymbol und $t_1, \ldots, t_k \in \mathcal{T}_\sigma$ dann gilt

$$[\![f(t_1, \ldots, t_k)]\!]^{\mathcal{I}} := f^{\mathcal{A}}\big([\![t_1]\!]^{\mathcal{I}}, \ldots, [\![t_k]\!]^{\mathcal{I}}\big). \qquad \dashv$$

Wir können nun die Semantik prädikatenlogischer Formeln definieren.

Definition 5.11 (Semantik der Prädikatenlogik) Sei σ eine Signatur. Induktiv über den Formelaufbau definieren wir eine Funktion $[\![\cdot]\!]^{\cdot}$, die jeder Formel $\varphi \in \text{FO}[\sigma]$ und jeder σ-Interpretation $\mathcal{I} := (\mathcal{A}, \beta)$ für φ einen Wahrheitswert $[\![\varphi]\!]^{\mathcal{I}} \in \{0, 1\}$ zuordnet.

Basisfall.

- Für alle Terme $t, t' \in \mathcal{T}_\sigma$ definieren wir

$$[\![t = t']\!]^{\mathcal{I}} := \begin{cases} 1 & \text{wenn } [\![t]\!]^{\mathcal{I}} = [\![t']\!]^{\mathcal{I}} \\ 0 & \text{sonst.} \end{cases}$$

- Für alle k-stelligen Relationssymbole $R \in \sigma$ und alle Terme $t_1, \ldots, t_k \in \mathcal{T}_\sigma$ definieren wir

$$[\![R(t_1, \ldots, t_k)]\!]^{\mathcal{I}} := \begin{cases} 1 & \text{wenn } ([\![t_1]\!]^{\mathcal{I}}, \ldots, [\![t_k]\!]^{\mathcal{I}}) \in R^{\mathcal{A}} \\ 0 & \text{sonst.} \end{cases}$$

Induktionsschritt.

- Die Semantik der Verknüpfungen $\neg, \vee, \wedge, \rightarrow, \leftrightarrow$ ist wie in der Aussagenlogik definiert, z.B. wenn $\varphi := \neg\psi \in \text{FO}[\sigma]$ dann definieren wir

$$[\![\varphi]\!]^{\mathcal{I}} := 1 - [\![\psi]\!]^{\mathcal{I}}.$$

- Wenn $\varphi := \exists x\psi \in \text{FO}[\sigma]$, dann definieren wir

$$[\![\varphi]\!]^{\mathcal{I}} := \begin{cases} 1 & \text{es gibt } a \in A, \text{ so dass } [\![\psi]\!]^{\mathcal{I}[x/a]} = 1 \\ 0 & \text{sonst.} \end{cases}$$

- Wenn $\varphi := \forall x\psi \in \text{FO}[\sigma]$, dann definieren wir

$$[\![\varphi]\!]^{\mathcal{I}} := \begin{cases} 1 & [\![\psi]\!]^{\mathcal{I}[x/a]} = 1 \text{ für alle } a \in A \\ 0 & \text{sonst.} \end{cases} \qquad \dashv$$

Wir betrachten als nächstes einige Beispiele von Formeln und deren Bedeutung.

Beispiel 5.12 Sei $\sigma := \{+, *, 0, 1\}$ die Signatur der Arithmetik und sei $\mathcal{A} := (\mathbb{N}, +^{\mathcal{A}}, *^{\mathcal{A}}, 0^{\mathcal{A}}, 1^{\mathcal{A}})$ die Struktur über den natürlichen Zahlen mit der üblichen Interpretation von $+^{\mathcal{A}}, *^{\mathcal{A}}, 0^{\mathcal{A}}, 1^{\mathcal{A}}$. Sei $\beta : x \mapsto 2, y \mapsto 3$ und $\mathcal{I} := (\mathcal{A}, \beta)$. Dann gilt

- $[\![x * x]\!]^{\mathcal{I}} := \beta(x) *^{\mathcal{A}} \beta(x) = 4$ und
- $[\![x * x = y + 1]\!]^{\mathcal{I}} = 1$, da $\beta(x) *^{\mathcal{A}} \beta(x) = \beta(y) +^{\mathcal{A}} 1^{\mathcal{A}} = 4$.

Sei $\varphi(x, y) := \exists z(x * x = y + z)$. Um zu zeigen, dass $[\![\varphi]\!]^{\mathcal{I}} = 1$, müssen wir ein Element $a \in \mathbb{N}$ finden mit $[\![x * x = y + z]\!]^{\mathcal{I}\cup\{z\mapsto a\}} = 1$. Sei $\beta' := \beta \cup \{z \mapsto 1\}$ und $\mathcal{I}' := (\mathcal{A}, \beta')$. Dann gilt $[\![x * x = y + z]\!]^{\mathcal{I}'} = 1$ wie zuvor und daher $[\![\exists z(x * x = y + z)]\!]^{\mathcal{I}} = 1$. ⊣

Beispiel 5.13 Sei $\sigma_{Graph} := \{E\}$.

- $\varphi := E(x, y)$. Dann ist $\mathit{frei}(\varphi) = \{x, y\}$. Sei $\mathcal{I} = (\mathcal{G}, \beta)$ mit $\mathcal{G} = (V, E^{\mathcal{G}})$, $\beta(x) := u \in V$ und $\beta(y) := v \in V$. Dann gilt $[\![\varphi]\!]^{\mathcal{I}} = 1$ genau dann, wenn es eine Kante zwischen u und v in $\mathcal{G}$ gibt.
- Sei $\varphi := \exists x \forall y\big(x = y \vee E(x, y)\big)$. Dann ist $\mathit{frei}(\varphi) := \varnothing$ und φ gilt in einem Graphen, wenn es einen Knoten mit Kanten zu allen anderen gibt.
- Sei $\varphi := \exists x_1 \exists x_2\big((E(x, x_1) \wedge E(x_1, x_2)) \wedge E(x_2, y)\big)$. Dann ist $\mathit{frei}(\varphi) := \{x, y\}$. Dann gilt φ in einem Graphen mit Belegung β, wenn es einen Weg der Länge 1 oder 3 von $\beta(x)$ zu $\beta(y)$ gibt. Man beachte, dass x, x_1, x_2 und y nicht unbedingt paarweise verschieden sein müssen.
- Sei $\varphi := \forall x \forall y\big(E(x, y) \to E(y, x)\big)$. Dann ist $\mathit{frei}(\varphi) = \varnothing$ und φ gilt in einem Graphen, wenn er „ungerichtet“ist, d.h. es zu jeder Kante von u nach v auch die Kante von v nach u gibt. ⊣

Beispiel 5.14 Sei $\sigma := \{<\}$ die Signatur der Ordnungen, d.h. $<$ ist eine binäre Relation. Wir beschreiben im folgenden einige Eigenschaften von Relationen $<$ durch logische Formeln.

(1) „$<$ ist irreflexiv“: $\forall x\, \neg x < x$.

(2) „$<$ ist transitiv“: $\forall x \forall y \forall z\big((x < y \wedge y < z) \to x < z\big)$.

(3) Das heißt, wir können wie folgt sagen, dass $<$ eine partielle Ordnung ist:

$$\varphi_{\textit{par-ord}} := \Big(\forall x\ \neg x < x\Big) \wedge \Big(\forall x \forall y \forall z \big((x < y \wedge y < z) \to x < z\big)\Big).$$

(4) „$<$ ist total“: $\varphi_t := \forall x \forall y (x < y \vee x = y \vee y < x)$.

(5) Eine lineare Ordung $<$ wird formalisiert durch $\varphi_{\textit{ord}} := \varphi_{\textit{par-ord}} \wedge \varphi_t$. ⊣

Beispiel 5.15 Ein *vertex cover* eines ungerichteten Graphen $\mathcal{G} := (V, E^{\mathcal{G}})$ ist eine Menge $X \subseteq V$, so dass $u \in X$ oder $v \in X$ für alle Kanten $e := \{u, v\} \in E^{\mathcal{G}}$. Das vertex cover Problem ist das folgende Problem: Gegeben ein Graph $\mathcal{G}$ und $k \in \mathbb{N}$, enthält $\mathcal{G}$ ein vertex cover der Größe $\leqslant k$?

In dieser Formalisierung wird eine Aussage über eine Menge X und über alle Kanten gemacht (nicht über Knoten). Dies können wir in der Prädikatenlogik nicht direkt ausdrücken. Wir formulieren das Problem daher um.

$\mathcal{G}$ enthält ein vertex cover der Größe $\leqslant k$, wenn

- es k Knoten $x_1, \ldots, x_k$ gibt, so dass
- für alle u, v: Wenn es eine Kante zwischen u und v gibt, dann ist u eins der x_i oder v eins der x_i.

Das können wir nun eins-zu-eins in die Prädikatenlogik übersetzen.

$$\exists x_1 \ldots \exists x_k\ \forall u \forall v \Big(E(u, v) \to \big(\bigvee_{i=1}^{k} u = x_i \vee \bigvee_{i=1}^{k} v = x_i\big)\Big)$$ ⊣

Wie in der Aussagenlogik hängt eine Formel nur von den Teilen einer Interpretation ab, die auch von der Formel benutzt werden.

Theorem 5.16 (Koinzidenzlemma) *Seien σ, τ, τ' Signaturen, so dass $\sigma \subseteq \tau \cap \tau'$. Sei $\mathcal{I} := (\mathcal{A}, \beta)$ eine τ-Interpretation und $\mathcal{J} := (\mathcal{B}, \gamma)$ eine τ'-Interpretation, so dass*

- *$A = B$,*
- *$S^{\mathcal{A}} = S^{\mathcal{B}}$ für alle Symbole S, die in σ vorkommen.*

(1) *Ist $t \in \mathcal{T}_\sigma$ ein σ-Term und $\beta(x) = \gamma(x)$ für alle $x \in \mathrm{var}(t)$, dann*

$$[\![t]\!]^{\mathcal{I}} = [\![t]\!]^{\mathcal{J}}.$$

(2) *Ist* $\varphi \in \text{FO}[\sigma]$ *eine Formel und* $\beta(x) = \gamma(x)$ *für alle* $x \in \mathit{frei}(\varphi)$*, dann*

$$\mathcal{I} \models \varphi \qquad \Longleftrightarrow \qquad \mathcal{J} \models \varphi.$$

Notation 5.17 (1) Sei $\varphi \in \text{FO}[\sigma])$ eine Formel mit $\mathit{frei}(\varphi) \subseteq \{x_1, \ldots, x_k\}$. Sei $\mathcal{A}$ eine σ-Struktur und β eine Belegung, so dass $\beta(x_i) := a_i$, für alle $1 \leqslant i \leqslant k$. Wir schreiben $\mathcal{A} \models \varphi[x_1/a_1, \ldots, x_k/a_k]$ statt $\mathcal{I} \models \varphi$. Dies ist möglich, da nach dem Koinzidenzlemma die Belegung der Variablen außer $x_1, \ldots, x_k$ nicht relevant ist.

(2) Erinnerung: Wir schreiben $\varphi(x_1, \ldots, x_k)$ um anzudeuten, dass $\mathit{frei}(\varphi) \subseteq \{x_1, \ldots, x_k\}$. In diesem Fall vereinfachen wir obige Notation weiter und schreiben $\mathcal{A} \models \varphi[a_1, \ldots, a_k]$. Man beachte, dass dies von der Sequenz $x_1, \ldots, x_k$ abhängt.

(3) Wenn φ ein Satz ist, schreiben wir $\mathcal{A} \models \varphi$. ⊣

5.3. Relationale Datenbanken

Eine *relationale Datenbank* ist eine endliche Menge von *„Tabellen“*. Z. B. könnte eine Filmdatenbank wie imdb.org wie folgt aussehen

Schauspieler

Schausp.	*ID*	*Geburtsdatum*
George Clooney	1	6. Mai 1961
Scarlett Johansson	2	22. November 1984
Jeff Daniels	3	19. Februar 1955
...	...	...

Filme

Titel	*Regie*	*Schau.*
Good night ... and good luck	Georg Clooney	1
Good night ... and good luck	Georg Clooney	3
Lost in translation	Sofia Coppola	2
...	...	...

Die Menge τ von *Tabellennamen* heißt *Datenbankschema*.

- Jede *Spalte* einer Tabelle in der Datenbank enthält Einträge vom selben Typ, z.B. Wörter oder Zahlen. In Datenbankterminologie werden Spaltennamen *Attribute* genannt. Jedes Attribut i hat einen Typ D_i, genannt *domain*.

- Jede *Zeile* der Tabelle enthält ein Tupel $(x_1, \ldots, x_n) \in D_1 \times D_2 \times \cdots \times D_n$.

Eine Datenbanktabelle kann daher als n-stellige Relation über der Menge $D := D_1 \cup \cdots \cup D_n$ aufgefasst werden. Eine relationale Datenbank mit Schema τ kann also als τ-Struktur $\mathcal{D}$ wie folgt geschrieben werden:

- Das Universum $A := D_1 \cup D_2 \cup \cdots \cup D_n$ ist die Vereinigung aller Domains.
- Für jede Tabelle $R \in \tau$ enthält die Struktur eine Relation $R^{\mathcal{D}}$, die alle Tupel der Tabelle enthält.

Relationale Datenbanken können also leicht als mathematische Strukturen modelliert werden. Eine Abfrage an eine Datenbank entspricht dann dem Auswerten logischer Formeln in Strukturen. Es existiert daher ein enger Zusammenhang zwischen mathematischer Logik, besonders dem Teilgebiet der *endlichen Modelltheorie*, und der Theorie relationaler Datenbanken. Historisch wurden relationale Datenbanken nach logischen Strukturen modelliert. Die Theorie relationaler Strukturen, insbesondere die Ausdrucksstärke von Logiken als Anfragesprache war bereits sehr gut untersucht und hat die Datenbanktheorie stark beeinflusst.

Beispiel 5.18 Betrachten wir noch einmal das Filmbeispiel. Der domain aller Einträge sind Zeichenketten. Sei Σ^* die Menge aller Zeichenketten über dem Alphabet $\{a, \ldots, z, A, \ldots, Z, 0, \ldots, 9\}$. Die Filmdatenbank entspricht folgender Struktur $\mathcal{D}$ über der Signatur $\sigma := \{\mathit{Actors}, \mathit{Movies}\}$:

- Das Universum ist $D := \Sigma^*$.
- Die Relationen
  - $\mathit{Actors}^{\mathcal{D}} := \{$ *(George Clooney, 1, 6 May 1961), (Scarlett Johansson, 2, 22 November 1984), (Jeff Daniels, 3, 19 February 1955)* $\}$ und
  - $\mathit{Movies}^{\mathcal{D}} :=$ { *(Good night ... and good luck, Georg Clooney, 1), (Good night ... and good luck, Georg Clooney, 3), (Lost in translation, Sofia Coppola, 2)* }. ⊣

Im obigen Beispiel ist das Universum der Struktur $\mathcal{D}$ unendlich. Dies ist eine unangenehme Eigenschaft, da die Komplemente von endlichen Relationen dann unendlich sind. Datenbanken werden daher meistens als *endliche Strukturen* modelliert. Das Universum enthält nur den *active domain* der Datenbank, d.h. die Menge aller Elemente, die in der Datenbank vorkommen.

Beispiel 5.19 Im Beispiel ist der active domain die Menge

D := { George Clooney, Scarlett Johansson, Jeff Daniels, 1, 2, 3,
6 May 1961, 22 November 1984, 19 February 1955, Good night ...
and good luck, Lost in translation, Sofia Coppola }.

Die Relationen sind wie zuvor definiert. Die Datenbank entspricht also der Struktur $\mathcal{D}$ über der Signatur $\sigma = \{Actors, Movies\}$ mit Universum D und den Relationen

- $Actors^{\mathcal{D}} := \{$ (George Clooney, 1, 6 May 1961), (Scarlett Johansson, 2, 22 November 1984), (Jeff Daniels, 3, 19 February 1955) (Nicole Kidman, 4, 20 June 1967) $\}$ und
- $Movies^{\mathcal{D}} := \{$ (Good night ... and good luck, Georg Clooney, 1), (Good night ... and good luck, Georg Clooney, 3), (Lost in translation, Sofia Coppola, 2) $\}$.

⊣

Beispiel 5.20 Mögliche Anfragen über der Signatur des obigen Beispiels sind:

(1) $\exists x \exists d \exists n_1 \exists n_2$ Movies$(x, d, n_1) \wedge$ Movies$(x, d, n_2) \wedge n_1 \neq n_2$, d.h.
„Es gibt einen Film mit mehr als einem Schauspieler“.

(2) $\exists d \exists f \exists n \exists b$ Movies$(f, d, n) \wedge$ Actors(d, n, b), d.h.
„Es gibt einen Regisseur, der im eigenen Film mitspielt.“ ⊣

Die Anfragen beschreiben Eigenschaften der Datenbank, die wahr oder falsch sein können. Derartige Anfragen werden als *Boolesche Anfragen* bezeichnet. Meistens wollen wir nicht-Boolesche Informationen aus einer Datenbank gewinnen, z.B. *“Gib alle Filme von Georg Clooney aus”.*

Dies können wir durch eine Anfrage mit freien Variablen erreichen:

$$\varphi(f) := \exists n \text{ Movies}(f, \text{„G-Clooney“}, n).$$

Hierbei ist *G-Clooney* ein Konstantensymbol, dass durch das Object „Georg Clooney“ interpretiert wird. Formal arbeiten wir also in Strukturen über der Signatur *{Actors, Movies, G-Clooney}*.

Definition 5.21 Sei $\mathcal{A}$ eine σ-Struktur und $\varphi(x_1, \ldots, x_k) \in \text{FO}[\sigma]$. Für $k \geqslant 1$ definieren wir

$$\varphi(\mathcal{A}) := \{(a_1, \ldots, a_k) \in A^k : \mathcal{A} \models \varphi[a_1, \ldots, a_k]\}. \qquad \dashv$$

Bemerkung 5.22 Die Relation $\varphi(\mathcal{A})$ hängt nicht nur von $\mathcal{A}$ sondern auch von der Sequenz $(x_1, \ldots, x_k) \in \text{VAR}^k$ ab. Wir müssen daher diese Sequenz jeweils angeben, bevor wir die Notation benutzen können. Diese Anforderung ist sinnvoll und findet sich in der Methodendefinition aller gängigen Programmiersprachen. Sei z.B.

`Boolean phi(int` x_1`, ..., int` x_k`)`

eine Java-Methode. Dann wird mit $x_1, \ldots, x_k$ eine Ordnung der Parameter festgelegt. Wir können dann *`Boolean b = phi(3, 5,..., 17);`* benutzen. $\dashv$

In praktischen Anwendungen von Datenbanken verwendet man meistens nicht die Prädikatenlogik as Anfragesprache. Die wichtigste Anfragesprache in relationalen Datenbanksystemen ist die *standard query language* (SQL). SQL ist äquivalent zur Prädikatenlogik in dem Sinn, dass sich genau die gleichen Eigenschaften und Anfragen an Datenbanken definieren lassen[1]. Allerdings enthält SQL noch weitere Funktionen, wie z.B. Aggregationsfunktionen (Mittelwerte, Summenbildung etc).

In SQL erhält man alle Filme von Regisseur George Clooney mit folgender Anfrage.

```
SELECT Title
FROM Movies
WHERE Director=''George Clooney''
```

Als Antwort erhält man die Tabelle

Answer	Good night ... and good luck

In der Sprache der relationalen Strukturen entspricht dies der unären Relation

$$\varphi((\mathcal{D}, \text{G-Clooney})) = \{\text{Good night} \ldots \text{and good luck}\}.$$

Es ist eins der eleganten Eigenschaften des relationalen Datenbankmodells, dass Antworten auf Anfragen selbst wieder Tabellen sind und somit in Datenbanken gespeichert werden können.

[1] Aktuelle SQL-Standards enthalten auch Konzepte zur Fixpunktbildung, die über die Prädikatenlogik hinausgehen. Allerdings werden diese selten verwendet, daher ist der logische Kern von SQL immernoch die Prädikatenlogik.