Relationale Algebra

Notation:

- $\mathcal{A}(e)$ Attribute der von e erzeugten Tupel (auch sch(e))
- $\mathcal{F}(e)$ Freie Variablen des Ausdrucks e
- $x_{|A}$ Die Projektion des Tupels x auf die Attribute A
- $x =_{|A} y$ Abgekürzte Schreibweise für $x_{|A} = y_{|A}$

Relationale Algebra:

$e_1 \cup e_2$	Vereinigung	
	Voraussetzung:	$\mathcal{A}(e_1) = \mathcal{A}(e_2)$
	Schema:	$\mathcal{A}(e_1)$
	Ausprägung:	$\{x \mid x \in e_1 \vee x \in e_2\}$
$e_1 \cap e_2$	Schnitt	
	Voraussetzung:	$\mathcal{A}(e_1) = \mathcal{A}(e_2)$
	Schema:	$\mathcal{A}(e_1)$
	Ausprägung:	$\{x \mid x \in e_1 \wedge x \in e_2\}$
$e_1 \setminus e_2$	Differenz	
	Voraussetzung:	$\mathcal{A}(e_1) = \mathcal{A}(e_2)$
	Schema:	$\mathcal{A}(e_1)$
	Ausprägung:	$\{x \mid x \in e_1 \wedge x \notin e_2\}$
$\rho_{a \to b}(e)$	Umbenennung	
	Voraussetzung:	$a \in \mathcal{A}(e) \wedge b \notin \mathcal{A}(e)$
	Schema:	$\mathcal{A}(e) \setminus \{a\} \cup \{b\}$
	Ausprägung:	$\{x_{\mid \mathcal{A}(e) \setminus \{a\}} \circ [b : x.a] \mid x \in e\}$
$\Pi_A(e)$	Projektion	
	Voraussetzung:	$A \subseteq \mathcal{A}(e)$
	Schema:	A
	Ausprägung:	$\{x_{\mid A} \mid x \in e\}$
$e_1 \times e_2$	Kreuzprodukt	
	Voraussetzung:	$\mathcal{A}(e_1) \cap \mathcal{A}(e_2) = \emptyset$
	Schema:	$\mathcal{A}(e_1) \cup \mathcal{A}(e_2)$
	Ausprägung:	$\{x \circ y \mid x \in e_1 \wedge y \in e_2\}$
$\sigma_p(e)$	Selektion	
	Voraussetzung:	$\mathcal{F}(p) \subseteq \mathcal{A}(e)$
	Schema:	$\mathcal{A}(e)$
	Ausprägung:	$\{x \mid x \in e \wedge p(x)\}$

Abgeleitet Operatoren:

$e_1 \bowtie_p e_2$	Join	
	Voraussetzung:	$(\mathcal{A}(e_1) \cap \mathcal{A}(e_2) = \emptyset) \wedge (\mathcal{F}(p) \subseteq (\mathcal{A}(e_1) \cup \mathcal{A}(e_2)))$
	Schema:	$\mathcal{A}(e_1) \cup \mathcal{A}(e_2)$
	Ausprägung:	$\{x \circ y \mid x \in e_1 \wedge y \in e_2 \wedge p(x \circ y)\}$
$e_1 \bowtie e_2$	Natürlicher Join	
	Voraussetzung:	$\forall_{[a:ta] \in \mathcal{A}(e_1), [b:tb] \in \mathcal{A}(e_2)} a \neq b \vee ta = tb$
	Schema:	$\mathcal{A}(e_1) \cup \mathcal{A}(e_2)$
	Ausprägung:	$\{x \circ y_{\mid \mathcal{A}(e_2) \setminus \mathcal{A}(e_1)} \mid x \in e_1 \wedge y \in e_2 \wedge x =_{\mid \mathcal{A}(e_1) \cap \mathcal{A}(e_2)} y\}$
$e_1 \div e_2$	Division	
	Voraussetzung:	$\mathcal{A}(e_2) \subseteq \mathcal{A}(e_1)$
	Schema:	$\mathcal{A}(e_1) \setminus \mathcal{A}(e_2)$
	Ausprägung:	$\{x_{\mid \mathcal{A}(e_1) \setminus \mathcal{A}(e_2)} \mid x \in e_1 \wedge \forall y \in e_2 : x =_{\mid \mathcal{A}(e_1) \cap \mathcal{A}(e_2)} y\}$
$e_1 \ltimes_p e_2$	Semijoin	
	Voraussetzung:	$(\mathcal{A}(e_1) \cap \mathcal{A}(e_2) = \emptyset) \wedge (\mathcal{F}(p) \subseteq (\mathcal{A}(e_1) \cup \mathcal{A}(e_2)))$
	Schema:	$\mathcal{A}(e_1)$
	Ausprägung:	$\{x \mid x \in e_1 \wedge \exists y \in e_2 : p(x \circ y)\}$
$e_1 \triangleright_p e_2$	Antijoin	
	Voraussetzung:	$(\mathcal{A}(e_1) \cap \mathcal{A}(e_2) = \emptyset) \wedge (\mathcal{F}(p) \subseteq (\mathcal{A}(e_1) \cup \mathcal{A}(e_2)))$
	Schema:	$\mathcal{A}(e_1)$
	Ausprägung:	$\{x \mid x \in e_1 \wedge \nexists y \in e_2 : p(x \circ y)\}$
$e_1 ⟕_p e_2$	outer-join	
	Voraussetzung:	$(\mathcal{A}(e_1) \cap \mathcal{A}(e_2) = \emptyset) \wedge (\mathcal{F}(p) \subseteq (\mathcal{A}(e_1) \cup \mathcal{A}(e_2)))$
	Schema:	$\mathcal{A}(e_1) \cup \mathcal{A}(e_2)$
	Ausprägung:	$(e_1 \bowtie_p e_2) \cup \{x \circ \circ_{a \in \mathcal{A}(e_2)} [a : NULL] \mid x \in (e_1 \triangleright_p e_2)\}$
$e_1 ⟗_p e_2$	full outer-join	
	Voraussetzung:	$(\mathcal{A}(e_1) \cap \mathcal{A}(e_2) = \emptyset) \wedge (\mathcal{F}(p) \subseteq (\mathcal{A}(e_1) \cup \mathcal{A}(e_2)))$
	Schema:	$\mathcal{A}(e_1) \cup \mathcal{A}(e_2)$
	Ausprägung:	$(e_1 ⟕_p e_2) \cup (e_2 ⟕_p e_1)$

Erweiterungen der relationalen Algebra:
(nicht Teil der ursprünglichen relationalen Algebra)

$\chi_{a:f}(e)$	map	
	Voraussetzung:	$a \notin \mathcal{A}(e) \wedge \mathcal{F}(f) \subseteq \mathcal{A}(e)$
	Schema:	$\mathcal{A}(e) \cup \{a\}$
	Ausprägung:	$\{x \circ [a : f(x)] \mid x \in e\}$
$\Gamma_{A;a:f}(e)$	Gruppierung	
	Voraussetzung:	$A \subseteq \mathcal{A}(e) \wedge a \notin A \wedge f$ is a function on $p \subseteq e$
	Schema:	$A \cup \{a\}$
	Ausprägung:	$\{x \circ [a : f(y)] \mid x \in \Pi_A(e) \wedge y = \{z \mid z \in e \wedge x =_{\mid A} z\}\}$
$e_1 ⋈_p e_2$	Abhängiger Join	
	Voraussetzung:	$(\mathcal{A}(e_1) \cap \mathcal{A}(e_2) = \emptyset) \wedge (\mathcal{F}(p) \subseteq (\mathcal{A}(e_1) \cup \mathcal{A}(e_2))) \wedge (\mathcal{F}(e_2) \subseteq \mathcal{A}(e_1))$
	Schema:	$\mathcal{A}(e_1) \cup \mathcal{A}(e_2)$
	Ausprägung:	$\{x \circ y \mid x \in e_1 \wedge y \in e_2(x) \wedge p(x \circ y)\}$

Formalisierung von Tupeln und Relationen:
(Dient nur zur Erläuterung, nicht für die Vorlesung notwendig)

Ein *Tupel* ist eine Zuordnung von Werten zu Attributnamen. Wenn ein Tupel t ein Attribut a enthält ($a \in t$) liefert $t.a$ den zugehörigen Wert. Induktiv ist ein Tupel wie folgt definiert:

1. Das leere Tupel $t = []$ ist ein Tupel. Es gilt $\forall a : a \notin t$.
2. Ein Tupel mit einem Attribut $t = [a : x]$ ist ein Tupel. Es gilt $a \in t \wedge \forall a'(a' \neq a \Rightarrow a' \notin t)$ sowie $t.a = x$.
3. Sind t_1 und t_2 disjunkte Tupel (d.h. $\forall a(a \notin t_1 \vee a \notin t_2)$) so ist auch die Konkatenation $t = t_1 \circ t_2$ ein Tupel. Es gilt $\forall a((a \in t_1 \Rightarrow a \in t \wedge t.a = t_1.a) \wedge (a \in t_2 \Rightarrow a \in t \wedge t.a = t_2.a) \wedge (a \notin t_1 \wedge a \notin t_2 \Rightarrow a \notin t))$.

Zwei Tupel t_1 und t_2 sind identisch ($t_1 = t_2$) genau dann wenn gilt:

$$\forall a((a \in t_1 \Rightarrow a \in t_2) \wedge (a \in t_2 \Rightarrow a \in t_1) \wedge (a \in t_1 \Rightarrow t_1.a = t_2.a)).$$

Ein *Schema* ist eine eine Zuordnung von Typen (Wertebereichen) zu Attributnamen, d.h. ein Schema ist ein Tupel bei dem alle Werte Mengen sind.
Ein Tupel t *entspricht* einem Schema S genau dann wenn gilt:

$$\forall a((a \in S \Rightarrow a \in t) \wedge (a \in t \Rightarrow a \in S) \wedge (a \in S \Rightarrow t.a \in S.a)).$$

Sofern die Typen aus dem Zusammenhang klar sind identifiziert man ein Schema häufig mit der Menge der enthaltenen Attribute. Entsprechend kann man (etwas informal) Mengenoperationen auf Schemata definieren:

- Wenn S_1 und S_2 Schemata sind so ist auch $S_1 \setminus S_2 = \{[x : S_1.x] | x \in S_1 \wedge x \notin S_2\}$ ein Schema.
- Wenn S_1 und S_2 Schemata sind so ist auch $S_1 \cap S_2 = \{[x : S_1.x] | x \in S_1 \wedge x \in S_2\}$ ein Schema. Hinweis: Diese Operation ist meist nur sinnvoll wenn S_1 und S_2 typkonform sind oder getestet werden soll ob S_1 und S_2 disjunkt sind.
- Zwei Schemata S_1 und S_2 sind *typkonform* genau dann wenn $\forall a(a \in S_1 \wedge a \in S_2 \Rightarrow S_1.a = S_2.a)$.
- Wenn S_1 und S_2 typkonforme Schemata sind so ist auch $S_1 \cup S_2 = S_1 \circ (S_2 \setminus (S_2 \cap S_1))$ ein Schema.
- Wenn S ein Schema ist so gilt $S = \emptyset$ genau denn wenn $S = []$. (Bei der ersten Gleichung wird S als die Menge seiner enthaltenen Attribute aufgefasst).

Hinweis: Alle oben definieren relationalen Operatoren gehen implizit davon aus dass alle beteiligten Schemata typkonform sind, d.h. der Attributname den Typ festlegt. Das gleiche gilt für freie Variablen.

Eine *Relation* R besteht aus einem Schema $\mathcal{R}$ und einer Instanz R. (Häufige Schreibweise: R (oder R) für die Instanz, $\mathcal{A}(R)$ (oder sch(R)) für das Schema).
$\mathcal{R}$ ist dabei ein beliebiges Schema, R eine Menge von Tupeln die dem Schema $\mathcal{R}$ entsprechen.