**Database Theory**
**Winter Term 2013/14**
Prof. Dr. W. May

# 2. Unit: Kalkül II

Discussion by 4./11.12.2013

**Exercise 6 (Division: Äquivalenz von Algebra und Kalkül)** For the relational algebra, the division operator has been introduced as a derived operator (cf. lecture "Databases"). Consider the relation schemata $r(A, B)$ and $s(B)$.

$$r \div s = \{\mu \in Tup(A) \mid \{\mu\} \times s \subseteq r\} = \pi[A](r) \setminus \pi[A]((\pi[A](r) \times s) \setminus r).$$

Derive a query in the relational calculus from the left-hand side, and prove the equivalence with the right-hand side.

---

The left-hand side expression: the set of all possible tuples over a $A$ is described by $F(X) = ADOM(X)$. The remaining task is then easy: for all values $Y$ in $S$, the combination of $X$ and $Y$ must be in $R$:

$$F(X) = ADOM(X) \wedge \forall Y : (s(Y) \rightarrow r(X, Y)) \ .$$

Here, it is obvious that instead $ADOM(X)$, the consideration can be restricted to the $A$-values of $R$:

$$F(X) = \exists Z : r(X, Z) \wedge \forall Y : (s(Y) \rightarrow r(X, Y)) \ .$$

The query is not in SRNF. It is equivalent to

$$F(X) = \exists Z : r(X, Z) \wedge \neg\exists Y : (s(Y) \wedge \neg r(X, Y)) \ ,$$

which is in SRNF (thus, domain-independent), but not in RANF.

Transformation to RANF ("push-into-not-exists"):

$$F(X) = \exists Z : r(X, Z) \wedge \neg\exists Y : (\exists Z_2 : r(X, Z_2)) \wedge s(Y) \wedge \neg r(X, Y))$$

Derivation of the algebra expression:

| $F$ | *Algebra* |
|---|---|
| $(\exists Z : r(X, Z)) \wedge s(Y) \wedge \neg r(X, Y)$ | $(\pi[A](r) \times s) \setminus r$ |
| $\exists Y : (\exists Z : r(X, Z)) \wedge s(Y) \wedge \neg r(X, Y)$ | $\pi[A]((\pi[A](r) \times s) \setminus r)$ |
| | (the expression has the format $A$) |
| $\exists Z : r(X, Z)$ | $\pi[A](r)$ (has again the format $A$) |
| $F(X)$ as above | $\pi[A](r) \setminus \pi[A]((\pi[A](r) \times s) \setminus r)$ |

... is exactly the right-hand side.

---

**Exercise 7 (Kalkül: Gruppierung und Aggregation)** Define a syntactical extension for the relational calulus, that allows to use aggregate functions similar to the `GROUP BY` functionality of SQL.

For this, consider only aggregate functions as simple applications over single attributes like max(population), but not more complex expressions like max(population/area).

- What is the result of an aggregate function, and how can it be used in the calculus?
- Which inputs does an aggregate function have?
- how can this input be obtained from the database?

Give a calculus expression for the query "For each country give the name and the total number of people living in its cities".

---

The result is a number. It can be bound to a variable or it can be used in a comparison. Thus, the aggregate function is to be considered as a term (whose evaluation yields a value).

The immediate input to an aggregate function is a set/list of values, over which the aggregate is computed (sum, count, ...).

This list can be obtained as results of a (sub)formula (similar to a correlated subquery) with a free variable.

The results are grouped by zero, one or more free variables of the subquery. Usually, these also occur in other literals outside the aggregation.

$$X = \textit{agg-op}\{\textit{var}\ [\textit{group-by-vars}]; \textit{subq-fml}\}$$

where in *subq-fml* the *group-by-vars* and *var* have free occurrences. E.g.,

$$\begin{array}{l} F(CN, SumCityPop) = \\ \quad \exists C, A, P, Cap, CapProv : country(CN, C, A, P, Cap, CapProv) \wedge \\ \quad SumCityPop = \mathsf{sum}\{CityPop\ [C]; \\ \qquad\qquad \exists CtyN, CtyProv, L1, L2 : city(CtyN, CtyProv, C, CityPop, L1, L2)\} \end{array}$$

groups by $C$, computes the sum over $CityPop$ and binds the value to $SumCityPop$.

Comments:

- a similar syntax is used in F-Logic;
- the usage in XSB is similar, but the user has to program it more explicitly:
  - the list is created by the Prolog predicate "bagof";
  - the aggregation operation over the list must be programmed in the common Prolog style for handling a list.

---

**Exercise 8 (Algebra→Kalkül)** Consider the relation schemata $R(A, B), S(B, C)$ und $T(A, B, C)$.

a) Give an equivalent safe calculus expression for the algebra expression

$$(\pi[A, B]((R \bowtie S) - T)) \cup R$$

b) Simplify it.

c) Give an equivalent safe calculus expresseion for the algebra expression

$$(\pi[A, B]((R \bowtie S) - T)) \cup \pi[A, B](\sigma[A < B](R) \bowtie T)$$

---

a) Proceed bottom-up (to know the variables that are already bound):

| $Algebra$ | $Calc$ | $free(\_)$ | $rr(\_)$ |
|---|---|---|---|
| $R \bowtie S$ | $R(A,B) \wedge S(B,C)$ | $A,B,C$ | $A,B,C$ |
| $(R \bowtie S) - T$ | $(R(A,B) \wedge S(B,C)) \wedge \neg S(A,B,C)$ | $A,B,C$ | $A,B,C$ |
| $\pi[A,B]((R \bowtie S) - T))$ | $\exists C : (R(A,B) \wedge S(B,C)) \wedge \neg T(A,B,C)$ | $A,B$ | $A,B$ |
| $(\pi[A,B]((R \bowtie S) - T))) \cup R$ | $(\exists C : (R(A,B) \wedge S(B,C)) \wedge \neg T(A,B,C)) \vee R(A,B)$ | $A,B$ | $A,B$ |

The expression is in SRNF and in RANF (and safe).

b) The expression is actually equivalent to $R$. The first part of the disjunction/union returns pairs $(A,B)$ from $R$, only with additional constraints.

c) The first part of the disjunction is the same as above. c Second part:

| $Algebra$ | $Calc$ | $free(\_)$ | $rr(\_)$ |
|---|---|---|---|
| $\sigma[A < B](R)$ | $R(A,B) \wedge A < B$ | $A,B$ | $A,B$ |
| $\sigma[A < B](R) \bowtie T$ | $R(A,B) \wedge A < B \wedge T(A,B,C)$ | $A,B,C$ | $A,B,C$ |
| $\pi[A,B](\sigma[A < B](R) \bowtie T)$ | $\exists C : (R(A,B) \wedge A < B \wedge T(A,B,C))$ | $A,B$ | $A,B$ |

Together:

$$\begin{aligned} F(A,B) = \; & (\exists C : (R(A,B) \wedge S(B,C)) \wedge \neg T(A,B,C)) \vee \\ & \exists C : (R(A,B) \wedge A < B \wedge T(A,B,C)) \end{aligned}$$

Note (Semijoin): second part of the disjunction is equivalent to $\sigma[A < B](R) \ltimes T$.
The corresponding calculus expression is $R(A,B) \wedge A < B \wedge \exists C : T(A,B,C))$.

---

**Exercise 9 (Kalkül→Algebra)** Consider the relation schemata $R(A,B), S(B,C)$ und $T(A,B,C)$.

a) Give an equivalent algebra expression for the following safe relational calculus expression:

$$F_1(X,Y) = T(Y,a,Y) \wedge (R(a,X) \vee S(X,c)) \wedge \neg T(a,X,Y)$$

Proceed as shown in the lecture for the equivalence proof.

b) Simplify the expression.

c) Extend the expression from 8a) to

$$F_2(Y) = \exists X : (F_1(X,Y) \wedge X > 3)$$

---

a) First, consider each of the three conjuncts (denoted as $F_2, F_1$ and $F_3$) separately:
The literal $F_1(Y) = T(Y,a,Y)$ corresponds to the subexpression

$$E_1 = \rho[A \to Y](\pi[A](\sigma[(A = C) \wedge (B = a)](T))) \; .$$

The subformula $F_2(X) = R(a,X) \vee S(X,c)$ corresponds to the expression

$$E_2 = \rho[B \to X](\pi[B](\sigma[A = a](R))) \cup \rho[B \to X](\pi[B](\sigma[C = c](S))) \; .$$

Negated literal $F_3(X,Y) = \neg T(a,X,Y)$: The literal $F_4(X,Y) = T(a,X,Y)$ corresponds to the expression

$$E_4 = \rho[B \to X, C \to Y](\pi[B,C](\sigma[A = a](T)))$$

According to the lecture, the expression corresponding to $F_3(X,Y)$ is then

$$E_3 = \rho[\$1 \to X, \$2 \to Y](ADOM^2) - DOM^2 - \rho[B \to X, C \to Y](\pi[B,C](\sigma[A = a](T)))$$

where $ADOM^2 = ((\pi[A](R) \cup \pi[B](R) \cup \pi[B](S) \cup \pi[C](S) \cup \pi[A](T) \cup \pi[B](T) \cup \pi[C](T)) \times (\pi[A](R) \cup \pi[B](R) \cup \pi[B](S) \cup \pi[C](S) \cup \pi[A](T) \cup \pi[B](T) \cup \pi[C](T)))$ contains all 2-tuples of values from the database.

Thus, $E = E_1 \bowtie E_2 \bowtie (ADOM^2 - E_4)$ is the complete algebra expression.

b) Simplify: $E_1$ and $E_2$ have no variable/column in common, thus it can be simplified as $(E_1 \times E_2) \bowtie (ADOM^2 - E_4)$. Both subterms bind $X$ and $Y$, thus, $ADOM^2$ can be omitted, obtaining $E' = (E_1 \times E_2) - E_4$.

c) The additional comparison is expressed as a selection, and the $\exists X$ quantification is expressed as a projection to $Y$:

$$\pi[Y](\sigma[X > 3](E'))$$