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

1. Unit: Kalkül I

Self-contained subformulas (i.e., formulas in RANF) will also be needed for translating complex queries into Datalog programs. The RANF section has not been discussed in detail, thus, solve the below exercises intuitively (i.e., find relational algebra expressions in the same way as you did it for the DB lecture before knowing the relational calculus at all).

Discussion by 15.5./20.5.2015

Exercise 1 (Kalkül: Sichere, Wertebereichsunabhängige Anfragen) Check for the following queries whether they are in SRNF (give $rr(G)$ for each of their subformulas).

For the formulas that are in RANF:

- check whether the formulas are in RANF. If not, give an equivalent formula in RANF.
- Give equivalent expressions in the relational algebra and in SQL (develop the SQL expressions both from the original formula and from the RANF formula).

a) $F(X,Y,Z) = p(X,Y) \wedge (q(Y) \vee r(Z))$,
b) $F(X,Y) = p(X,Y) \wedge (q(Y) \vee r(X))$,
c) $F(X,Y) = p(X,Y) \wedge \neg\exists Z : r(Y,Z)$,
d) $F(X) = p(X) \wedge \exists Y : (q(Y) \wedge \neg r(X,Y))$,
e) $F(X) = p(X) \wedge \neg\exists Y : (q(Y) \wedge \neg r(X,Y))$
f) $F(X,Y) = \exists V : (r(V,X) \wedge \neg s(X,Y,V)) \wedge \exists W : (r(W,Y) \wedge \neg s(Y,X,W))$

---

a) $p(X,Y) \wedge (q(Y) \vee r(Z))$:

G	$rr(G)$
$p(x,y)$	X,Y
$q(Y)$	Y
$r(Z)$	Z
$q(Y) \vee r(Z)$	$\{Y\} \cap \{Z\} = \emptyset$
$p(X,Y) \wedge (q(Y) \vee r(Z))$	$\{X,Y\} \cup \emptyset = \{X,Y\}$

Since $free(F) = \{X,Y,Z\} \neq \{X,Y\} = rr(F)$, F is not in SRNF (and thus also not in RANF). F is not domain-independent: for $\mathcal{S}$ with $\mathcal{S}(p) = \{1,a\}$ and $\mathcal{S}(q) = \{(a)\}$ and $\mathcal{S}(r) = \emptyset$ and domain $\mathcal{D}$ is the answer set $\{X \mapsto 1, Y \mapsto a, Z \mapsto d | d \in \mathcal{D}\}$.

b) $p(X,Y) \wedge (q(Y) \vee r(X))$:

G	$rr(G)$
$p(x,y)$	X,Y
$q(Y)$	Y
$r(X)$	X
$q(Y) \vee r(X)$	$\{Y\} \cap \{X\} = \emptyset$
$p(X,Y) \wedge (q(Y) \vee r(X))$	$\{X,Y\} \cup \emptyset = \{X,Y\}$

Since $free(F) = \{X, Y\} = rr(F)$, F is in SRNF.
F is not in RANF since the disjunction $q(Y) \vee r(X)$ is not self-contained.
F can easily be expressed in SQL (with $P(P_1, P_2), Q(Q_1), R(R_1)$):

```
SELECT P1,P2
FROM P
WHERE P2 in (SELECT Q1 FROM Q)
  OR P1 in (SELECT R1 FROM R)
```

The equivalent expression in the relational algebra is
$(P \bowtie_{P_2=Q_1} Q) \cup (P \bowtie_{P_1=R_1} R)$.
This is also obtained when translating from SRNF to RANF with "push-into-or":
$(p(X, Y) \wedge q(Y)) \vee (p(X, Y) \wedge r(Z))$
and then translates as usual to the relational algebra.

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

G	$rr(G)$
$p(X, Y)$	X, Y
$r(Y, Z)$	Y, Z
$\exists Z : r(Y, Z)$	Y
$\neg\exists Z : r(Y, Z)$	$\emptyset$
$p(X, Y) \wedge \neg\exists Z : r(Y, Z)$	X, Y

Since $free(F) = \{X\} = rr(F)$, F is in SRNF.
All subformulas are self-contained.
F can easily be expressed in SQL (with $P(P_1, P_2), R(R_1, R_2)$):

```
SELECT P1,P2
FROM P
WHERE P2 NOT IN (SELECT R2 FROM R)
```

The equivalent expression in the relational algebra is
$P \bowtie (\pi[P_2](P) - \pi[R_1](R))$.
The standard translation that uses the enumeration formula for the active domain (here: those that occur in P and R) reads as:
$P \bowtie ((\pi[P_1](P) \cup \pi[P_2](P) \cup \pi[R_1](R) \cup \pi[R_2](R)) - \pi[R_1](R))$.

d) $F(X) = p(X) \wedge \exists Y : (q(Y) \wedge \neg r(X, Y))$:

G	$rr(G)$
$p(X)$	X
$q(Y)$	Y
$r(X, Y)$	X, Y
$\neg r(X, Y)$	$\emptyset$
$q(Y) \wedge \neg r(X, Y)$	Y
$\exists Y : q(Y) \wedge \neg r(X, Y)$	$\emptyset$
$p(X) \wedge \exists Y : q(Y) \wedge \neg r(X, Y)$	X

Since $free(F) = \{X\} = rr(F)$, F is in SRNF.
F is not in RANF since the subformula $G = \exists Y : q(Y) \wedge \neg r(X, Y)$ is not self-contained: for the body $H = q(Y) \wedge \neg r(X, Y)$ there is $free(H) = \{X, Y\} \supsetneq \{Y\} = rr(H)$ (note that the SAFE criterion from the lecture would already detect H as the problem).
F can easily be expressed in SQL (with $P(P_1), Q(Q_1), R(R_1, R_2)$):

```
SELECT P1
FROM P
```

```
WHERE EXISTS (SELECT Q1
              FROM Q
              WHERE (P1,Q1) NOT IN (SELECT R1,R2 FROM R))
```

The equivalent expression in the relational algebra is
$\pi[P_1]((P \times Q) - \rho[R_1 \to P_1, R_2 \to P_2]R$.
This is also obtained when translating from SRNF to RANF with "push-into-exist":
$F(X) = \exists Y : (p(X) \wedge q(Y) \wedge \neg r(X,Y))$,
and then translates as usual to the relational algebra.
This corresponds to the (simpler) SQL query

```
SELECT P1
FROM P, Q
WHERE (P1,Q1) NOT IN (SELECT R1,R2 FROM R)
```

e) This formula is the pattern of the relational division, $r \div q$. Es ist äquivalent zu $F(X) = p(X) \wedge \forall Y : (q(Y) \to r(X,Y))$.
$F(X) = p(X) \wedge \neg\exists Y : (q(Y) \wedge \neg r(X,Y))$,

G	$rr(G)$
$p(X)$	X
$q(Y)$	Y
$r(X,Y)$	X,Y
$\neg r(X,Y)$	$\emptyset$
$q(Y) \wedge \neg r(X,Y)$	Y
$\exists Y : q(Y) \wedge \neg r(X,Y)$	$\emptyset$
$\neg\exists Y : q(Y) \wedge \neg r(X,Y)$	$\emptyset$
$p(X) \wedge \neg\exists Y : q(Y) \wedge \neg r(X,Y)$	X

Since $free(F) = \{X\} = rr(F)$, F is in SRNF.
F is –as in (d)– not in RANF since the subformula $G = \exists Y : q(Y) \wedge \neg r(X,Y)$ is not self-contained.
F can easily be expressed in SQL (with $P(P_1), Q(Q_1), R(R_1, R_2)$):

```
SELECT P1
FROM P
WHERE NOT EXISTS (SELECT Q1
                  FROM Q
                  WHERE (P1,Q1) NOT IN (SELECT R1,R2 FROM R))
```

The equivalent expression in the relational algebra is
$P - \pi[P_1]((P \times Q) - \rho[R_1 \to P_1, R_2 \to P_2](R))$.
This is also obtained when translating from SRNF to RANF with "push-into-not-exist":
$F(X) = p(X) \wedge \neg\exists Y : (p(X) \wedge q(Y) \wedge \neg r(X,Y))$,
and then translates as usual to the relational algebra.

f) This is an example for a conjunction, where none of the conjuncts is self-contained:

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

G	$rr(G)$
$r(V,X)$	X,V
$s(X,Y,V)$	X,Y,V
$\neg s(X,Y,V)$	$\emptyset$
$r(V,X) \wedge \neg s(X,Y,V)$	X,V
$\exists V : (r(V,X) \wedge \neg s(X,Y,V))$	X
$r(W,Y)$	W,Y
$s(Y,X,W)$	X,Y,W
$\neg s(Y,X,W)$	$\emptyset$
$r(W,Y) \wedge \neg s(Y,X,W)$	W,Y
$\exists W : (r(W,Y) \wedge \neg s(Y,X,W))$	Y
$(\ldots) \wedge (\ldots)$	X,Y

Since $free(F) = \{X,Y\} = rr(F)$, F is in SRNF.
F is not in RANF since the subformulas $\exists V : (r(V,X) \wedge \neg s(X,Y,V))$ and $\exists W : (r(W,Y) \wedge \neg s(Y,X,W))$ are not self-contained (again, the problem is located inside each of the subformulas, as SAFE would complain about).
F can easily be expressed in SQL (with $P(P_1), Q(Q_1), R(R_1,R_2)$):

```
SELECT rv.R2, rw.R2
FROM R rv, R rw
WHERE NOT EXISTS (SELECT * FROM S
                  WHERE S1=rv.R2 and S2=rw.R2 and S3=rv.R1)
  AND NOT EXISTS (SELECT * FROM S
                  WHERE S1=rw.R2 and S2=rv.R2 and S3=rw.R1)
```

or

```
SELECT rv.R2, rw.R2
FROM R rv, R rw
WHERE NOT (rv.R2, rw.R2, rv.R1 IN (SELECT * FROM S))
  AND NOT (rw.R2, rv.R2, rw.R1 IN (SELECT * FROM S))
```

The equivalent expression in the relational algebra is ... not that easy.
Thus, F has to be transformed from SRNF to RANF by moving the first conjunct into the second by "push-into-exists" (or the vice versa, the final result is the same):

$$\exists W : \exists V : (r(V,X) \wedge \neg s(X,Y,V)) \wedge (r(W,Y) \wedge \neg s(Y,X,W))$$

Flatten existential quantifiers, flatten conjunction:

$$\exists V,W : B(X,Y,V,W)$$

with $B = (r(V,X) \wedge r(W,Y) \wedge \neg s(X,Y,V) \wedge \neg s(Y,X,W))$
is self-contained with $free(B) = \{V,W,X,Y\} = rr(B)$
According to the transformation algorithm given in the lecture, the following has to be done:

- build the (XYV) component of B, subtract s,
- in parallel build the (XYW) component of B, subtract s,
- these are the triples of bindings that "survive",
- join them,

- and project to:

$$\begin{aligned}\pi[X,Y](\quad & (\pi[X,Y,V](\rho[R_1 \to V, R_2 \to X](r) \times \rho[R_1 \to W, R_2 \to Y](r)) \\ & -\rho[S_1 \to X, S_2 \to Y, R_2 \to V](s)) \\ \bowtie\ & (\pi[X,Y,W](\rho[R_1 \to V, R_2 \to X](r) \times \rho[R_1 \to W, R_2 \to Y](r)) \\ & -\rho[S_1 \to Y, S_2 \to X, R_2 \to W](s)))\end{aligned}$$

Exercise 2 (Relationale Anfragen an Mondial: Schweizer Sprachen) Give expressions in the relational calculus for the following queries against the Mondial database. Compare with the same queries in the relational Algebra and in SQL.

a) All codes of countries, in which some languages is spoken that is also spoken in Switzerland.

b) All codes of countries, in which only languages are spoken that are not spoken in Switzerland.

c) All codes of countries, in which only languages are spoken that are also spoken in Switzerland.

d) All codes of countries in which all languages that are spoken in Switzerland are also spoken.

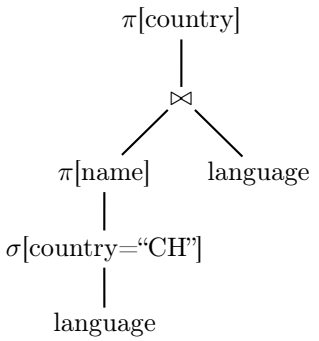
---

a) $F(C) = \exists L, Perc1, Perc2 : (\text{language}('CH', L, Perc1) \wedge \text{language}(C, L, Perc2))$

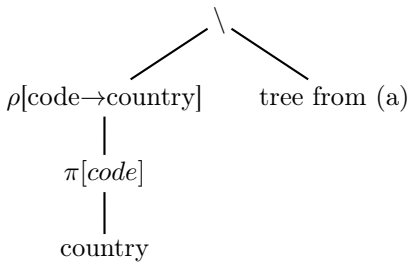
b) $F(C) = \exists CN, A, Pop, Cap, CapP :$
$(\text{country}(CN, C, A, Pop, Cap, CapP) \wedge$
$\neg\exists L, Perc1, Perc2 : (\text{language}('CH', L, Perc1) \wedge \text{language}(C, L, Perc2)))$

Algebra:

a)



(b)



c) $F(C) = (\exists CN, A, Pop, Cap, CapP : \text{country}(CN, C, A, Pop, Cap, CapP)) \wedge$
$\neg\exists L, Perc1 : (\text{language}(C, L, Perc1) \wedge \neg\exists Perc1 : \text{language}('CH', L, Perc2))$

d) $F(C) = (\exists CN, A, Pop, Cap, CapP : \text{country}(CN, C, A, Pop, Cap, CapP)) \wedge$
$\forall L : ((\exists Perc1 : \text{language}('CH', L, Perc1)) \rightarrow (\exists Perc2 : \text{language}(C, L, Perc2)))$

In the discussion, also the Datalog queries have been discussed:

```
:- auto_table.
:- include(mondial).

aufgA(C) :- language(C,_X,_), language('CH',_X,_).
aufgB(C) :- country(_,C,_,_,_,_), not aufgA(C).
nonCHLgCtry(C) :- language(C,_L,_), not language('CH',_L,_).
onlyCHLgCtry(C) :- country(_,C,_,_,_,_), not nonCHLgCtry(C).
chLgMissing(C) :- country(_,C,_,_,_,_), language('CH',_L,_), not language(C,_L,_).
noCHLgMissing(C) :- country(_,C,_,_,_,_), not chLgMissing(C).
?- aufgA(C).
?- aufgB(C).
?- onlyCHLgCtry(C).   % note: also countries with no language!
?- noCHLgMissing(C).
```