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

# 1. Unit: Kalkül I

Discussion by 9./14./15.11.2013

**Exercise 1 (FOL: Tautologie)** Prove that the formula

$$F = (\forall x : (p(x) \rightarrow q(x)) \wedge \forall y : (q(y) \rightarrow r(y))) \rightarrow \forall z : (p(x) \rightarrow r(x)))$$

is a tautology.

First, convinve yourself that it is a tautology, and then prove it in formally.

---

Intutitively, it is clear: if for every $x$ where $p(x)$ holds, also $q(x)$ holds, and whenever $q(x)$ holds, then $r(x)$ holds, then it follows directly that whenever $p(x)$, $r(x)$ holds.

There are several ways for a formal proof.

- the first "option", namely investigating each structure that has unary predicates $p$, $q$ and $r$ (and maybe many others) is impossible, because one would have to check infinitely many structures. So one needs an argumentation that deals with families of structures.
- "Traditional" mathematical forward-proof:
  Consider any arbitrary structure $\mathcal{S} = (I, \mathcal{D})$ and prove that $\mathcal{S} \models F$. (i.e., we know nothing special about $\mathcal{S}$, but have to prove something about it ...).
  In the following, use abbreviations:

  $$F = (\underbrace{\underbrace{\forall x : (p(x) \rightarrow q(x))}_{F_{11}} \wedge \underbrace{\forall y : (q(y) \rightarrow r(y))}_{F_{12}})}_{F_1} \rightarrow \underbrace{\forall z : (p(x) \rightarrow r(x))}_{F_2})$$

  By definition of the semantics of "$\rightarrow$",

  $$\mathcal{S} models F_1 \rightarrow F_2 \Leftrightarrow \mathcal{S} models \neg F_1 \text{ or } \mathcal{S} \models F_2.$$

  So, $F$ holds trivially in all structures $\mathcal{S}$ that do *not* satisfy $F_1$.
  Thus, work has to be done for structures $\mathcal{S}$ that satisfy $F_1$ (now, we know something about the structures that have to be considered).
  Exploit this knowledge: $\mathcal{S} \models F_1$ which means $\mathcal{S} \models_\emptyset F_1$ (empty variable assignment, since there are no free variables) which in turn means $\mathcal{S} \models_\emptyset F_{11} \wedge F_{12}$.
  Lifting the logical symbol "$\wedge$" to the mathematical proof level, there is
  $\mathcal{S} \models_\emptyset F_{11}$ and $\mathcal{S} \models_\emptyset F_{12}$, i.e.
  $\mathcal{S} \models_\emptyset \forall x : (p(x) \rightarrow q(x))$ and $\mathcal{S} \models_\emptyset \forall y : (q(y) \rightarrow r(y))$.
  Next, lift the universal quantifier symbol to the mathematical level:
  For all $d \in \mathcal{D}$, $\mathcal{S} \models_{\emptyset^d_x} p(x) \rightarrow q(x)$ and for all $e \in \mathcal{D}$ $\mathcal{S} \models_{\emptyset^e_y} q(y) \rightarrow r(y)$.
  Next, lift the implications to the mathematical level:
  For all $d \in \mathcal{D}$, $\mathcal{S} \models_{\emptyset^d_x} \neg p(x)$ or $\mathcal{S} \models_{\emptyset^d_x} q(x)$ and For all $e \in \mathcal{D}$, $\mathcal{S} \models_{\emptyset^e_y} \neg q(y)$ or $\mathcal{S} \models_{\emptyset^e_y} q(y)$.
  Exploit the definition of $\models$ for atomic predicates:

(1) For all $d \in \mathcal{D}$, $d \notin I(p)$ or $d \in I(q)$ and

(2) For all $e \in \mathcal{D}$, $e \notin I(q)$ or $e \in I(r)$.

So, for each element $a \in \mathcal{D}$, $d \notin I(p)$ (3), or if $d \in I(p)$, then from (1) $a \in I(q)$. From (2), which states that either $a \notin I(q)$ (which is not the case for our $a$) or $a \in I(r)$ (4). Taking (3) and (4), is

Try tableau for $G \models F$:
Start with $G \wedge \neg F$.

$$\forall v : (p(v) \vee q(v)) \wedge \neg((\forall x : p(x)) \vee (\forall x : q(x))) \quad (1)$$
$$\forall v : (p(v) \vee q(v)) \quad (2)$$
$$p(V) \vee q(V) \quad (3)$$
$$\neg((\forall x : p(x)) \vee (\forall x : q(x))) \quad (4)$$
$$\neg(\forall x : p(x)) \wedge \neg(\forall x : q(x)) \quad (5)$$
$$\neg(\forall x : p(x)) \quad (6)$$
$$\neg(\forall x : q(x)) \quad (7)$$
$$\neg p(a) \quad (6)$$
$$\neg q(b) \quad (6)$$

now resolve (3):

$$\begin{array}{ccc} / & & \backslash \\ p(V) & & q(V) \\ \Box\ \{V/a\} & \dashrightarrow & q(a) \end{array}$$

(2) nochmal bereitstellen:

$$p(W) \vee q(W)$$
$$\begin{array}{ccc} / & & \backslash \\ p(W) & & q(W) \\ p(b) & \dashleftarrow & \Box\ \{W/b\} \end{array}$$

---

**Exercise 2 (FOL: Beweise)** a) Prove the equivalence of the expressions $\forall x : F(x)$ and $\neg\exists y : \neg F(y)$ by using the definition of the semantics of the formulas.

b) Prove: if in some application, there is a causal relationship between a formula $\varphi$ and another formula $\psi$ (i.e., if for any state where $\varphi$ holds, also $\psi$ is satisfied), then also the *material implication* $\varphi \rightarrow \psi$ holds in every state that satisfies the specification.

c) Give an example where the formula $\varphi \rightarrow \psi$ is satisfied without any causal relationship between them.

d) Give examples, where the material implication $\varphi \rightarrow \psi$ occurs in a reasonable way in a formula.

---

a) The subsequent (very explicit and detailed) solution focusses in the formal distinction between the three levels:
   - inside of the FOL syntax: $\neg$, $\forall$, $\exists$,
   - statements on the mathematical level *about* the formula (describing the relationship between an Interpretation $\mathcal{I}$ and the formula): $\models$, $\not\models$,
   - the “natural” meta level: “there are”, “for all”, “not”.

For all interpretations $\mathcal{I} = (I, \mathcal{D})$ and alle variable assignments $\beta$,

| | | |
|---|---|---|
| | $\mathcal{I} \models_\beta \forall x : F(x)$ | |
| $\Leftrightarrow$ | for all $d \in \mathcal{D}$, $\mathcal{I} \models_{\beta_x^d} F(x)$ | by definition of $\forall$ |
| $\Leftrightarrow$ | there is no $d \in \mathcal{D}$ such that not $\mathcal{I} \models_{\beta_x^d} F(x)$ | Step on the metalevel (talking about satisfying a formula) |
| $\Leftrightarrow$ | there is no $d \in \mathcal{D}$ such that $\mathcal{I} \not\models_{\beta_x^d} F(x)$ | Step on the math. level (talking about the formula) |
| $\Leftrightarrow$ | es gibt kein $d \in \mathcal{D}$, so dass $\mathcal{I} \models_{\beta_x^d} \neg F(x)$ | Step into the FOL formula by definition of $\neg$ |
| $\Leftrightarrow$ | not (there is some $d \in \mathcal{D}$ such that $\mathcal{I} \models_{\beta_x^d} F(x)$) | Step on the metalevel |
| $\Leftrightarrow$ | not $\mathcal{I} \models_\beta \exists x : F(x)$ | by definition of $\exists$ |
| $\Leftrightarrow$ | $\mathcal{I} \not\models_\beta \exists x : F(x)$ | Step on the math. level (talking about the formula) |
| $\Leftrightarrow$ | $\mathcal{I} \models_\beta \neg\exists x : F(x)$ | by definition of $\neg$ |
| $\Leftrightarrow$ | $\mathcal{I} \models_\beta \neg\exists y : F(y)$ | ... and rename the variable |

Consequences:

- Proof that both formulas are euivalent, i.e., can be replaced by each other,
- proof that the semantics assigned to the FOL syntax is defined as it is intuitively understood by humans.

b) Consider an structure $\mathcal{I}$ that represents such a situation in some application. The application guarantees that if $\mathcal{I} \models \varphi$ is the case, then also $\mathcal{I} \models \psi$. We show that then the material implication $\mathcal{I} \models \varphi \rightarrow \psi$ is satisfied.

$$\begin{array}{ll} & \mathcal{I} \models \varphi \rightarrow \psi \\ \Leftrightarrow & \mathcal{I} \models \neg\varphi \vee \psi \\ \Leftrightarrow & \mathcal{I} \models \neg\varphi \text{ oder } \mathcal{I} \models \psi \end{array}$$

Case distinction: if $\mathcal{I} \models \neg\varphi$, then "yes"; otherwise, i.e., $\mathcal{I} \models \varphi$, then by causality, $\mathcal{I} \models \psi$, and again "yes".

Thus, $\mathcal{I} \models \varphi \rightarrow \psi$ holds in any such structure.

c) Consider an interpretation $\mathcal{I}$ of some simplified signature of the Mondial database that holds the current geographical facts.

$$\mathcal{I} \models \mathsf{encompassed}(\text{"Germany"},\text{"Europe"}) \rightarrow \mathsf{borders}(\text{"BR"},\text{"RA"})$$

("if in $\mathcal{I}$ the fact that (i) Germany belons to Europe is contained, then also the fact that (ii) Brasil and Argentina are neighbors is true". (i): yes. (ii): yes. – the formula is satisfied.)

In the same way,

$$\mathcal{I} \models \mathsf{encompassed}(4, \text{"North Sea"}) \rightarrow \mathsf{borders}(\text{"BR"},\text{"RA"})$$

("if in $\mathcal{I}$ the pair (i) (4, North Sea) is contained in the "encompassed" relation (which is not the case since the tuple is rubbish), then, also the tuple (ii) (BR,RA) is contained in the "borders" relation." (i) no. (ii) yes - but this does not matter at all.)

Analogously

$$\mathcal{I} \models \mathsf{encompassed}(4, \text{"North Sea"}) \rightarrow \mathsf{borders}(\text{"Berlin"}, \text{"04.05.2012"})$$

$\neg A \vee B$ – if $\neg A$ holds, $B$ isn't relevant at all.

(Note: in logic there is no datatype check – the domain contains things, numbers, strings etc.)

- The most important usage, as long as only a single state is considered, but no general resoning about states is done, is to require some condition $\psi$ only for a restricted subset of the domain that is specified by $\varphi$: "if $x$ satisfies $\varphi$, then it is required also to satisfy $\psi$".

– Integrity constraints can be expressed by universally quantified material implications, e.g., $\forall x : (\exists y : \textsf{encompassed}(x, y)) \rightarrow (\exists n, p, a, cap, cp : \textsf{country}(n, x, p, a, cap, cp))$.
"if a value $x$ occurs in "encompassed" (as country code), then there must be a tuple in "country" with that code."
If some state does not satisfy this formula, then there is some tuple in "encompassed" that contains a country code that does not exist in "country" as key value.
For the query

$$F(x) = (\exists y : \textsf{encompassed}(x, y)) \wedge \neg(\exists n, p, a, cap, cp : \textsf{country}(n, x, p, a, cap, cp))$$

there must be no answer bindings for $x$.
The negated form is used as *denials*: the formula

$$\neg\exists x : (\exists y : \textsf{encompassed}(x, y)) \wedge \neg(\exists n, p, a, cap, cp : \textsf{country}(n, x, p, a, cap, cp))$$

must be satisfied in all database states.

– The relational division also uses the material implication:

$$F(Y) = (\exists X : R(X, Y)) \wedge \forall X : (S(X) \rightarrow R(X, Y))$$

$R(X, Y)$ is only required for those $X$ that are in $S(X)$.

---

**Exercise 3 (Kalkül: Sichere, Wertebereichsunabhängige Anfragen)** Check for the following queries whether they are in SRNF (give $rr(G)$ for each of their subformulas).

Check also, 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(P1,P2), Q(Q1), R(R1)):

```
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
$(P \bowtie [P2 = Q1]Q) \cup (P \bowtie [P1 = R1]R)$.
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(P1,P2), R(R1,R2)):

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

The equivalent expression in the relational algebra is
$P \bowtie (\pi[P2](P) - \pi[R1](R))$.
The standard translation that uses the enumeration formula for the active domain (here: those that occur in P and R) reads as:
$P \bowtie ((\pi[P1](P) \cup \pi[P2](P) \cup \pi[R1](R) \cup \pi[R2](R)) - \pi[R1](R))$.

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(P1), Q(Q1), R(R1,R2)):

```
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
$\pi[P1]((P \times Q) - \rho[R1 \to P1, R2 \to P2]R$.
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:
$F(X) = p(X) \wedge \neg\exists Y : (q(Y) \wedge \neg r(X,Y))$,

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

Since $free(F) = \{X\} = rr(F)$, $F$ is in SRNF.
$F$ is –as in (c)– 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(P1), Q(Q1), R(R1,R2)):

```
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
$P - \pi[P1]((P \times Q) - \rho[R1 \to P1, R2 \to P2](R))$.
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(P1), Q(Q1), R(R1,R2)):

```
SELECT rv.R2, r2.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, r2.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 $(XYV)$ componente 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:

$$\begin{aligned}\pi[X,Y]( \quad & (\pi[X,Y,V](\rho[R1 \to V, R2 \to X](r) \times \rho[R1 \to W, R2 \to Y](r)) \\ & \quad -\rho[S1 \to X, S2 \to Y, R2 \to V](s)) \\ & \bowtie (\pi[X,Y,W](\rho[R1 \to V, R2 \to X](r) \times \rho[R1 \to W, R2 \to Y](r)) \\ & \quad -\rho[S1 \to Y, S2 \to X, R2 \to W](s)))\end{aligned}$$

---

**Exercise 4 (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.

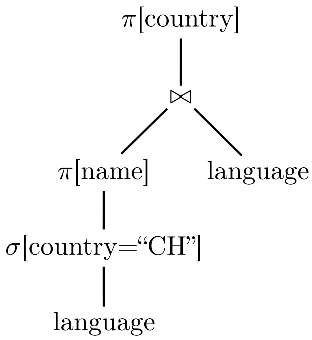
d) All codes of countries in which all languages that are spoken in Switzerland are also spoken.

---

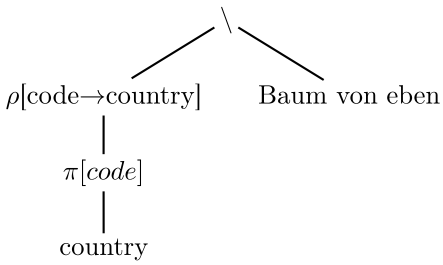
a) $F(C) = \exists L, Perc1, Perc2 : (\text{language}('CH', L, Perc1) \wedge \text{language}(C, L, Perc2))$

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)




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

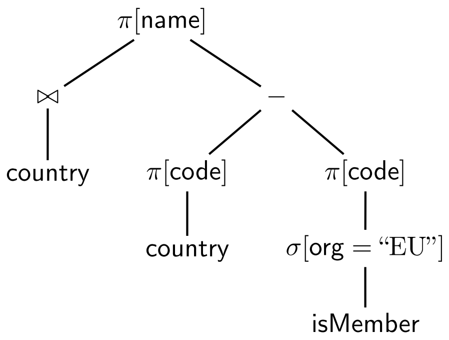
---

**Exercise 5 (RANF to Algebra – Minus)** Give expressions in the relational algebra and in the relational calculus for the query *"Full names of all countries that have more than 1000000 inhabitants and are not member of the EU"*.

Check whether the calculus expression is in SRNF and RANF, and transform it into the relational algebra. Compare the result with the algebra expression.

---

A straightforward algebra expression is




The calculus expression is

$$F(N) = \exists C, Cap, CapProv, A, Pop : \\ (\mathsf{country}(N, C, Cap, CapProv, A, Pop) \wedge Pop > 1000000 \wedge \neg\exists T : \mathsf{isMember}(C, \text{“EU”}, T))\ .$$

It is in SRNF, it is safe range, and it is in RANF. Recall that for the subformula $\neg\exists T : \mathsf{isMember}(C, \text{“EU”}, T)$, RANF requires $rr(\exists T : \mathsf{isMember}(C, \text{“EU”}, T)) = free(\exists T : \mathsf{isMember}(C, \text{“EU”}, T)) = \{C\}$ which is the case.

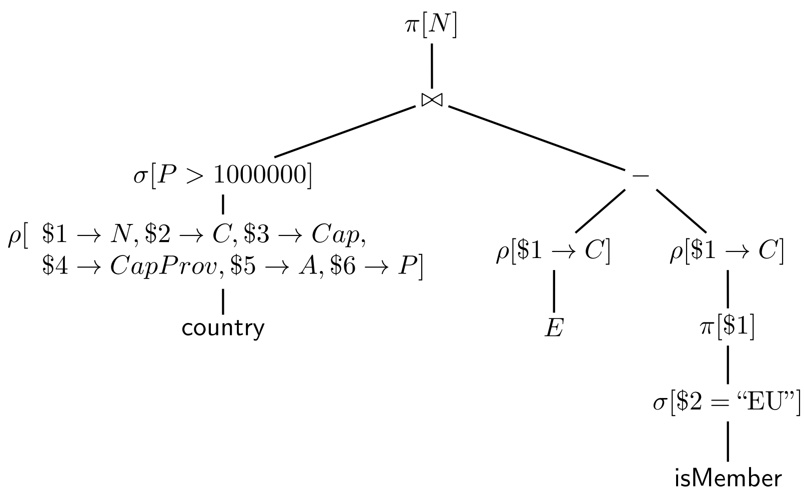
For the relational algebra,

$$\begin{array}{lcl} \mathsf{isMember}(C, \text{“EU”}, T) & \Rightarrow & \rho[\$1 \to C, \$3 \to T](\pi[\$1, \$3](\sigma[\$2 = \text{“EU”}](\mathsf{isMember}))) \\ \exists T : \mathsf{isMember}(C, \text{“EU”}, T) & \Rightarrow & \pi[\$1](\rho[\$1 \to C, \$3 \to T](\pi[\$1, \$3](\sigma[\$2 = \text{“EU”}](\mathsf{isMember})))) \\ & = & \rho[\$1 \to C](\pi[\$1](\sigma[\$2 = \text{“EU”}](\mathsf{isMember}))) \end{array}$$

For $\neg\exists T : \mathsf{isMember}(C, \text{“EU”}, T)$, let the expression $E$ denote the algebra expression that enumerates all values of the active domain. With this,

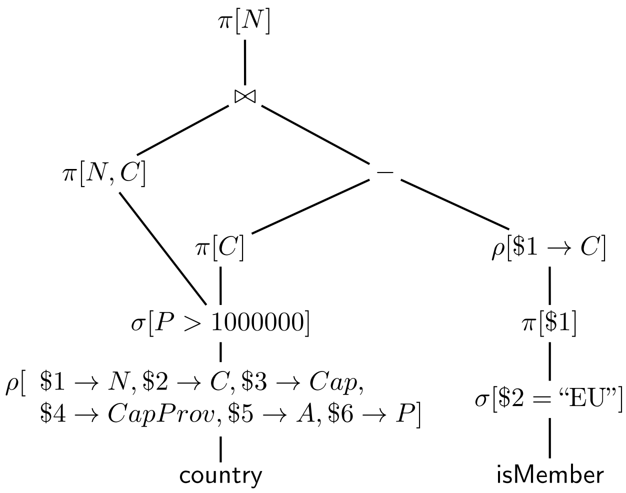
$$\neg\exists T : \mathsf{isMember}(C, \text{“EU”}, T) \Rightarrow \rho[\$1 \to C](E) - \rho[\$1 \to C](\pi[\$1](\sigma[\$2 = \text{“EU”}](\mathsf{isMember})))$$

Altogether, the whole query translates to




Obviously, the term $\rho[\$1 \to C](E)$ can be replaced by $\rho[\$2 \to C](\pi[\$2](\mathsf{country}))$ which enumerates a superset of all values of $C$ that can result from the left subtree.

Instead, also $\rho[\$2 \to C](\pi[\$2](\sigma[\$6 > 1000000](\mathsf{country})))$ is sufficient, which makes the left subtree (nearly) unnecessary. From it, only the full name must still be obtained.




Another possibility is the anti-join $\triangleright$ (which is one of the built-in operators of internal algebras):