BDD Cours 8 : Traduire TRC en SQL grâce aux lois d'équivalences logiques

C. Kuttler

18 octobre 2016

Nous traduisons TRC tel que présenté en cours la semaine dernière (avec implication et quantification universelle et existentielle) en SQL. Pour cela nous utilisons trois lois d'équivalence logique (notation pour d'équivalence : $\equiv$). Elles permettent de transformer une formule TRC en forme normale, qui utilisant uniquement la quantification existentielle et la conjonction logique. Cette forme normale de TRC peut être directement réécrite en SQL. Rappel des les lois d'équivalence logique, déjà vues en première année de licence :

1. définition de l'implication
$$A \Rightarrow B \equiv (\text{ not } A) \text{ or } B \tag{1}$$
2. définition de la quantification universelle, en passant par la quantification existentielle et la négation :
$$\forall x : P(x) \equiv \text{ not } \exists x : \text{ not } P(x) \tag{2}$$
3. Lois de de Morgan
$$\text{not } (A \text{ and } B) \equiv \text{ not } A \text{ or not } B \tag{3}$$
$$\text{not } (A \text{ or } B) \equiv \text{ not } A \text{ and not } B \tag{4}$$

1 Portée maximale d'un avion

Problème 1. Quelle est la portée maximale d'un avion ?

— la portée d'un avion tel que pour tous les avions, la portée est égale ou inférieure
— la portée d'un avion tel qu'il N'existe PAS d'avion, tel que NOT(la portée est égale ou inférieure)
— la portée d'un avion telle qu'il n'existe pas d'avion, dont la portée est supérieure

$$\{t | \exists a \in avions : t[portee] = a[portee]$$
$$\text{and } \forall a2 \in avions : a2[portee] \leq a[portee]\}$$
$$\equiv$$

(élimination de $\forall$, equation 2, puis reecriture de not $\leq$ par $>$)

$$\{t | \exists a \in avions : t[portee] = a[portee]$$
$$\text{and not } \exists a2 \in avions : a2[portee] > a[portee]\}$$

Cette dernière version se réécrit directement en SQL, prenant en compte que les quantificateurs du premier niveau correspondent a la clause FROM, et les quantificateurs imbriqués aux $EXISTS$ imbriqués :

Requête 1.

```
select t.portee
from avions t
where not exists (select * from avions a2 where a2.portee>t.portee);
```

Remarque : Nous pouvons obtenir le même résultat, avec une requête SQL équivalente utilisant une sous-requête avec ALL :

Requête 2.

```
select t.portee
from avions t
where t.portee >=ALL (select portee from avions);
```

Regardons la tentative d'un débutant en SQL, qui démarre de la manière la plus simple d'obtenir la portée maximale d'un avion :

Requête 3.

```
select max(portee) from avions;
```

Dans ce cas, il faut imbriquer la requête initiale, pour obtenir l'identifiant de l'avion avec cette portée maximale :

Requête 4.

```
select aid from avions
where portee in (select max(portee) from avions);
```

2 Comment trouver le pilote maxivalent ?

Problème 2. Nous cherchons des pilotes certifié(s) pour TOUS les avions

- un pilote (eid) pour qui, pour tous les avions (aid), il existe une certification(aid,eid)
- un pilote (eid) pour qui, il n'existe pas d'avion (aid), tel qu'il n'existe pas de certification(aid,eid)

$$\{t | \exists e \in employes : \ t[enom] = e[enom] \text{ and } \forall a \in avions : \exists c \in certifications : c[aid] = a[aid] \text{ and } c[eid] = e[eid]\}$$

$$\equiv$$

élimination de la quantification universelle

$$\{t | \exists e \in employes \ : t[enom] = e[enom] \text{ and not } \exists a \in avions : \text{not } \exists c \in certifications : c[aid] = a[aid] \text{ and } e[eid] = c[eid]\}$$

Requête 5.

```
select t.enom
from employes t
where not exists (
   select * from avions a
   where not exists (
      select * from certifications c  where c.aid=a.aid and t.eid=c.eid));
```

Donc nous avons exprimé en SQL, comment identifier un pilote qualifié pour tous les avions en passant par deux sous-requêtes NOT EXISTS, qui correspondent en TRC à deux quantifications existentielles sous contrôle d'une négation. Ce qui est équivalent à une quantification du genre *pour tous*. Quel détails supplémentaire faut-il encore ajouter dans la requête ci-haut, et pourquoi ?

3 Vendeur pas bon marché

Problème 3. Nous cherchons des vendeurs offrant uniquement des articles à plus de 100 € Nous pouvons exprimer cela de différentes manières équivalentes :

1. vendeur dont tous les articles coûtent plus de 100 €. C'est facile a dire et comprendre, mais pour etre précis, il faut dire :
2. un vendeur, tel que pour tous les articles, si l'article est offert par ce vendeur, alors son prix est >100 €
3. vendeur qui n'offre pas d'article à ≤100 €
4. vendeur pour qui il n'existe pas d'article à ≤100 €

Remarque : Il faudrait également assurer que le vendeur offre au moins un article à plus de 100 €. Les formules suivantes ne contiennent pas cette condition, pour simplifier.

Voila la phrase (2) exprimée en TRC :

$$\{t | \exists f \in fournisseur : t[fnom] = f[fnom]$$
$$\forall c \in catalogue : c[fid] = f[fid] \Rightarrow c[prix] > 100\}$$
$$\equiv$$

élimination de la quantification universelle

$$\{t | \exists f \in fournisseur : t[fnom] = f[fnom]$$
$$\text{not } \exists c \in catalogue : \text{ not } \Big(c[fid] = f[fid] \Rightarrow c[prix] > 100\Big)\}$$
$$\equiv$$

élimination de l'implication

$$\{t | \exists f \in fournisseur : t[fnom] = f[fnom]$$
$$\text{not } \exists c \in catalogue : \underbrace{\text{not}}_{\text{pousser vers l'intérieur}} \Big(c[fid] \neq f[fid] \text{ or } c[prix] > 100\Big)\}$$
$$\equiv$$

de Morgan, puis détails :

$$\{t | \exists f \in fournisseur : t[fnom] = f[fnom]$$
$$\text{not } \exists c \in catalogue : (c[fid] = f[fid] \text{ and } c[prix] \leq 100)\}$$

Cette dernière version correspond à la phrase (4), et contrairement à (2), peut être réécrite en SQL comme suit :

Requête 6.
```
select t.fnom
from fournisseurs t
where not exists
   (select * from catalogue c where c.fid=t.fid and c.prix <=100);
```

Remarque : cette requête rend des fournisseurs n'ayant pas d'articles à moins de 100 €, même si ces vendeurs n'offrent aucun article. Il faut vérifier que le fournisseur offre également (au moins) un article à plus de 100 €. On peut le faire explicitement en ajoutant une deuxième condition existentielle :

Requête 7.
```
select t.fnom
from fournisseurs t
where not exists
    (select * from catalogue c where c.fid=t.fid and c.prix <=100)
and exists
    (select * from catalogue c where c.fid=t.fid and c.prix >100)
;
```

Ou, plus simplement :

Requête 8.
```
select t.fnom
from fournisseurs t, catalogue c
where c.fid=t.fid and c.prix>100 and
    not exists
    (select * from catalogue c2 where c2.fid=t.fid and c2.prix <=100);
```

4 Vendeur n'offrant aucun article vert

Problème 4. Nous cherchons un vendeur (ou des vendeurs), qui n'offre(nt) pas d'articles verts

- tous les articles offerts par ce vendeur sont d'une autre couleur que vert
- pour tous les articles du catalogue, si cet article est offert par ce fournisseur, alors cet article n'est pas vert
- il n'existe pas d'article de ce vendeur qui soit vert

La deuxieme phrase correspond a :

$$\{t | \exists f \in fournisseurs : f[fnom] = t[fnom] and \\ \forall c \in catalogue, \\ \forall a \in articles : \Big(c[fid] = f[fid] \text{ and } a[aid] = c[aid]\Big) \Rightarrow a[acoul] \neq' vert'\}$$

$$\equiv$$

élimination de la quantification universelle $\forall c \in catalogue$

$$\{t | \exists f \in fournisseurs : f[fnom] = t[fnom] : \\ \text{not } \exists c \in catalogue : \\ \underbrace{\text{not}}_{\text{prochaine réécriture}} \left(\forall a \in articles : \Big(c[fid] = f[fid] \text{ and } a[aid] = c[aid]\Big) \Rightarrow a[acoul] \neq' vert'\right)\}$$

$$\equiv$$

avec la variante de notre règle pour l'élimination du pour tous (un NOT en amont) $NOT\ \forall x : P(x) \equiv \exists x : notP(x)$

$$\{t|\exists f \in fournisseurs : f[fnom] = t[fnom] : \text{not } \exists c \in catalogue, (\exists a \in articles : NOT(\Big(c[fid] = f[fid] \text{ and } a[aid] = c[aid]\Big) \underbrace{\Rightarrow}_{\text{prochaine réécriture}} a[acoul] \neq' vert')\}$$

$$\equiv$$

réécriture de l'implication

$$\{t|\exists f \in fournisseurs : f[fnom] = t[fnom] : \text{not } \exists c \in catalogue, (\exists a \in articles : \underbrace{NOT}_{\text{prochaine réécriture}} (NOT\Big(c[fid] = f[fid] \text{ and } a[aid] = c[aid]\Big) \text{ or } a[acoul] \neq' vert')\}$$

$$\equiv$$

de Morgan

$$\{t|\exists f \in fournisseurs : f[fnom] = t[fnom] : \text{not } \exists c \in catalogue, (\exists a \in articles : (\text{not } \text{ not } \Big(c[fid] = f[fid] \text{ and } a[aid] = c[aid]\Big) \text{ and } \text{ not } a[acoul] \neq' vert')\}$$

suppression de la double négation, remplacement de (not $\neq$) par $=$

$$\{t|\exists f \in fournisseurs : f[fnom] = t[fnom] : \text{not } \exists c \in catalogue, (\exists a \in articles : (\Big(c[fid] = f[fid] \text{ and } a[aid] = c[aid]\Big) \text{ and } a[acoul] =' vert')\}$$

La dèrnière version se réécrit directement en SQL :

Requête 9.
```
select t.fnom
from fournisseurs t
where not exists (
   select * from catalogue c
   where exists (
      select * from articles a
      where c.fid=t.fid and a.aid=c.aid and a.acoul='vert'
)
);
```

qu'on peut simplifier en :

Requête 10.
```
select t.fnom
from fournisseurs t
where not exists(
      select * from catalogue c, articles a
      where a.aid=c.aid and c.fid=t.fid and a.acoul='vert'
);
```

5 Le cas inverse

Finalement nous cherchons l'inverse du problèmes précédent :

Problème 5. Comment identifier le vendeur qui offre uniquement des articles verts ?
- — vendeur offrant uniquement des articles verts
- — si un article est vendu par ce fournisseur, il est vert
- — un fournisseur tel que, pour tous les articles, si cet article est vendu par ce fournisseur, il est vert
- — un fournisseur qui ne vend pas d'articles dont la couleur n'est pas verte

$$\{t | \exists f \in fournisseur : t[fnom] = f[fnom] \text{ and } \forall c \in catalogue, \forall a \in articles : c[fid] = f[fid] \text{ and } a[aid] = c[aid] \Rightarrow a[acoul] =' vert'\}$$
$$\equiv$$

élimination de la quantification universelle

$$\{t | \exists f \in fournisseur : t[fnom] = f[fnom] \text{ and not } \exists c \in catalogue, \text{ not } (\forall a \in articles : c[fid] = f[fid] \text{ and } a[aid] = c[aid] \Rightarrow a[acoul] =' vert')\}$$
$$\equiv$$
$$\{t | \exists f \in fournisseur : t[fnom] = f[fnom] \text{ and not } \exists c \in catalogue, (\exists a \in articles : \text{ not } (c[fid] = f[fid] \text{ and } a[aid] = c[aid] \Rightarrow a[acoul] =' vert'))\}$$
$$\equiv$$

implication

$$\{t | \exists f \in fournisseur : t[fnom] = f[fnom] \text{ and not } \exists c \in catalogue, (\exists a \in articles : \text{ not } (\text{ not } (c[fid] = f[fid] \text{ and } a[aid] = c[aid]) \text{ or } a[acoul] =' vert'))\}$$
$$\equiv$$

de Morgan, éliminer la double négation

$$\{t | \exists f \in fournisseur : t[fnom] = f[fnom] \text{ and not } \exists c \in catalogue, (\exists a \in articles : c[fid] = f[fid] \text{ and } a[aid] = c[aid] \text{ and } a[acoul] \neq' vert')\}$$

Requête 11.
```
select t.fnom
from fournisseurs t
where not exists (
   select * from catalogue c
   where exists (
      select * from articles a
      where c.fid=t.fid and a.aid=c.aid and a.acoul<>'vert')
                          );
```

version directe et intuitive : il ne vend pas d'article qui n'est pas vert

Requête 12.
```
select t.fnom
from fournisseurs t
where not exists
   (select * from catalogue c, articles a
    where c.aid=a.aid and c.fid=t.fid and a.acoul<>'vert'
);
```

Même remarque que pour l'exemple 3. Il faut assurer que le vendeur offre au moins un article vert, dans la requête. Autrement, on obtient le fournisseur qui n'a pas d'article dans le catalogue. Nous avons omis cette condition en TRC pour simplifier.