Introduction au calcul relationnel

Jef Wijsen

Université de Mons (UMONS)

2 octobre 2020

Motivation

$$sort(ABUS) = \{Nom, Cru, Annee\}$$
$$sort(CRU) = \{Cru, Millesime, Qualite\}$$

$$\{z \mid \exists x \exists y \, (ABUS(\text{‘An’}, x, y) \wedge CRU(x, y, z))\}$$

Logique des prédicats (syntaxe)

Alphabet :

- des noms de relation (R, S, T) avec leurs arités
- des variables (x, y, z)

Formules :

- Si R est un nom de relation d'arité n et $x_1, x_2, \ldots, x_n$ sont des variables, alors $R(x_1, x_2, \ldots, x_n)$ est une formule dans laquelle les variables $x_1, \ldots, x_n$ sont libres.
- Si x est une variable libre d'une formule φ, alors $\exists x\varphi$ et $\forall x\varphi$ sont des formules dans lesquelles x n'est plus libre.
- Si φ est une formule, alors $\neg\varphi$ est une formule avec les mêmes variables libres que φ.
- Si φ_1 et φ_2 sont des formules, alors $\varphi_1 \wedge \varphi_2$, $\varphi_1 \vee \varphi_2$, $\varphi_1 \to \varphi_2$ et $\varphi_1 \leftrightarrow \varphi_2$ sont des formules ayant comme variables libres chaque variable qui est libre dans φ_1 ou φ_2.

Des parenthèses sont utilisées au besoin. Par simplicité, dans un premier temps, nous considérons seulement des formules sans égalité $(=)$ et sans constantes.

Formule avec indication des variables libres

Soit R une relation d'arité 2 et S une relation d'arité 1.

$$\underbrace{\exists y \left(\underbrace{R(x,y)}_{x,y} \wedge \forall z \underbrace{\left(\underbrace{R(x,z)}_{x,z} \rightarrow \underbrace{S(z)}_{z} \right)}_{x,z} \right)}_{x}$$

Logique des prédicats : sémantique "classique"

Une structure relationnelle est un couple $(\mathcal{D}, \mathcal{I})$ où $\mathcal{D}$ est un domaine fini (mais voir aussi slide 16 !) et non-vide et $\mathcal{I}$ associe une relation à chaque nom de relation. Ces relations utilisent seulement les valeurs de $\mathcal{D}$.
Une formule sans variables libres est soit **vraie**, soit **fausse** par rapport à cette structure.

Example 1 (Vérité d'une formule par rapport à une structure)

$$\mathcal{D} = \{a, e, i, o, u, b, c, d\}$$

$\mathcal{I}$

R	A	B
	a	b
	e	b
	a	c
	i	c

S	A
	a
	e
	o

$$\exists y\,((\exists x R(x, y)) \wedge \forall z\,(R(z, y) \rightarrow S(z)))$$

Si l'on choisit $y = b$, alors $\exists x R(x, \text{“}b\text{”})$ est vrai et $\forall z\,(R(z, \text{“}b\text{”}) \rightarrow S(z))$ est vrai. En conclusion, la formule est vraie par rapport à cette structure.

Example 2

$$\mathcal{D} = \{a, e, o, b\}$$

R	A	B
	a	b
	e	b

S	A
	a
	o

$$\exists y\,((\exists x R(x, y)) \wedge \forall z\,(R(z, y) \rightarrow S(z)))$$

La formule est fausse par rapport à cette nouvelle structure.

Observation : La vérité peut dépendre du domaine

Example 3

$$\mathcal{D} = \{a, e, i\} \qquad \begin{array}{c|c} S & A \\ \hline & a \\ & e \end{array}$$

$$\exists x\,(\neg S(x))$$

La formule est vraie par rapport à cette structure.

Example 4

$$\mathcal{D} = \{a, e\} \qquad \begin{array}{c|c} S & A \\ \hline & a \\ & e \end{array}$$

$$\exists x\,(\neg S(x))$$

La formule est fausse par rapport à cette nouvelle structure.

Observation : Tautologies et contradictions

Tautologie

$\exists x\,(S(x) \vee \neg S(x))$ est vrai par rapport à chaque structure.

Contradiction

$\exists x\,(S(x) \wedge \neg S(x))$ est faux par rapport à chaque structure.

Observation : la négation d'une tautologie est fausse par rapport à chaque structure (et est donc une contradiction).

Logique des prédicats : sémantique en bases de données

Une [instance de] base de données relationnelle $\mathcal{I}$ associe une relation à chaque nom de relation.

Le domaine reste inconnu ! ! ! Mais il est su que les valeurs dans les relations appartiennent au domaine.

Une formule est *d☹main-dependent* (et considérée comme étant erronée) si sa vérité peut dépendre du domaine.

Example 5

$$\mathcal{D} \supseteq \{a, e, i, o, b, c\}$$

R	A	B
	a	b
	e	b
	a	c
	i	c

S	A
	a
	e
	o

$$\exists y\,((\exists x R(x, y)) \wedge \forall z\,(R(z, y) \rightarrow S(z)))$$

Malgré que $\mathcal{D}$ soit [largement] inconnu, on peut conclure que la formule est vraie par rapport à cette structure.

Example 6

$$\mathcal{D} \supseteq \{a, e\}$$

S
a
e

$$\exists x\,(\neg S(x))$$

Cette formule est *d☹main-dependent* (et donc erronée en théorie des bases de données), parce que sa vérité dépend de $\mathcal{D}$. En effet, si $\mathcal{D} = \{a, e\}$, la formule est fausse ; si $\mathcal{D} = \{a, e, i\}$, la formule est vraie.

Noter :

- Si le domaine reste inconnu, il n'est pas toujours possible de déterminer la valeur de vérité (vrai ou faux) d'une formule qui est *d☹main-dependent*.
  "Eronné" n'est pas une troisième valeur de vérité !
- Une formule qui n'est pas *d☹main-dependent* est appelée *d☺main-independent* (et *correct*).

Au moins une espèce de fleur est disponible en toute couleur...

Fleur	*F*	*C*
	rose	rose
	rose	jaune
	tulipe	jaune

Couleur	*C*
	rose
	jaune
	brun

$$\exists x \left(\underbrace{\left(\exists y' \mathit{Fleur}(x, y')\right)}_{x \text{ est une fleur}} \wedge \forall y \left(\underbrace{\left(\exists x' \mathit{Fleur}(x', y)\right)}_{y \text{ est une couleur de fleur}} \rightarrow \mathit{Fleur}(x, y) \right) \right) \text{ est vrai}$$

$$\exists x \left(\underbrace{\left(\exists y' \mathit{Fleur}(x, y')\right)}_{x \text{ est une fleur}} \wedge \forall y \left(\underbrace{\mathit{Couleur}(y)}_{y \text{ est une couleur}} \rightarrow \mathit{Fleur}(x, y) \right) \right) \text{ est faux}$$

$$\exists x \left(\underbrace{\left(\exists y' \mathit{Fleur}(x, y')\right)}_{x \text{ est une fleur}} \wedge \forall y \left(\mathit{Fleur}(x, y) \right) \right) \text{ est } d\text{☹}\mathit{main\text{-}dependent}$$

Simplification (à mon goût personnel)

$$\exists x \left(\underbrace{\left(\exists y' \, Fleur(x, y')\right)}_{x \text{ est une fleur}} \wedge \forall y \left(\underbrace{\left(\exists x' \, Fleur(x', y)\right)}_{y \text{ est une couleur de fleur}} \rightarrow Fleur(x, y) \right) \right)$$

est équivalent à

$$\exists x \left(\underbrace{\left(\exists y' \, Fleur(x, y')\right)}_{x \text{ est une fleur}} \wedge \forall x' \forall y \left(Fleur(x', y) \rightarrow Fleur(x, y) \right) \right)$$

$$\begin{aligned}
\forall y \left(\left(\exists x' \left(F(x', y)\right)\right) \rightarrow F(x, y) \right) &\equiv \forall y \left(\neg \left(\exists x' \left(F(x', y)\right)\right) \vee F(x, y) \right) \\
&\equiv \forall y \left(\left(\forall x' \left(\neg F(x', y)\right)\right) \vee F(x, y) \right) \\
&\equiv \forall y \left(\forall x' \left(\neg F(x', y) \vee F(x, y)\right) \right) \\
&\equiv \forall x' \left(\forall y \left(\neg F(x', y) \vee F(x, y)\right) \right) \\
&\equiv \forall x' \left(\forall y \left(F(x', y) \rightarrow F(x, y)\right) \right)
\end{aligned}$$

Formules équivalentes

Soit φ une formule sans variables libres.
La notation $(\mathcal{D}, \mathcal{I}) \models \varphi$ est une manière succincte de dire "φ est vrai par rapport à la structure relationnelle $(\mathcal{D}, \mathcal{I})$".

Noter : si φ est *d☺main-independent*, on peut se contenter d'écrire $\mathcal{I} \models \varphi$.

Soient φ_1 et φ_2 deux formules, sans variables libres. Ces deux formules sont équivalentes (notation : $\varphi_1 \equiv \varphi_2$) si pour toute structure relationnelle $(\mathcal{D}, \mathcal{I})$, nous avons $(\mathcal{D}, \mathcal{I}) \models \varphi_1 \leftrightarrow \varphi_2$.

Theorem (Trakhtenbrot 1950)

There exists no algorithm for the following problem :

INPUT : *Two formulas φ_1, φ_2 without free variables.*

QUESTION : *Are φ_1 and φ_2 equivalent ?*

Trakhtenbrot et Turing (pour votre culture générale)

- Le théorème de Trakhtenbrot suppose que tout domaine soit fini. Noter : avec un nombre fini de valeurs, il est impossible de construire une relation infinie.
- En 1936, Turing avait prouvé un tel résultat pour des relations qui peuvent être finies ou infinies.
- Aucun de ces deux théorèmes implique l'autre ! ! !
- En fait, il existent des formules φ et ψ sans variables libres telles que :
  - par rapport à chaque relation finie, φ et ψ ont la même valeur de vérité ; mais
  - il existe une relation infinie par rapport à laquelle φ est vrai et ψ est faux.

Motivation pour la théorie des modèles finis

Soit $sort(R) = \{A, B\}$.

$$\begin{aligned} \text{PRIMARY KEY(A)} \quad & (1) \\ \text{UNIQUE(B)} \quad & (2) \\ \text{FOREIGN KEY(B) REFS R} \quad & (3) \\ \exists x\,((\exists y\ R(x,y)) \wedge \neg\exists z\ R(z,x)) \quad & (4) \end{aligned}$$

R	$\underline{A}$	B
	0	1
	1	2
	2	3
	⋮	⋮

un ensemble infini de tuples

$$\begin{aligned} \text{PK(A)} &= \forall x \forall y \forall z\,(R(x,y) \wedge R(x,z) \rightarrow y = z) \\ \text{UNIQUE(B)} &= \forall x \forall y \forall z\,(R(y,x) \wedge R(z,x) \rightarrow y = z) \\ \text{FK(B) REFS R} &= \forall x \forall y\,(R(x,y) \rightarrow \exists z\ R(y,z)) \end{aligned}$$

Vérifiez : Aucune relation finie respecte toutes ces quatre contraintes !

Donc, contrairement au théorème de Turing, le théorème de Trakhtenbrot considère que les formules

$$(1) \wedge (2) \wedge (3) \wedge (4) \text{ et } \exists x \exists y\,(R(x,y) \wedge \neg R(x,y))$$

sont équivalentes (il s'agit de deux contradictions).

La notion de domaine en théorie des bases de données

En théorie des bases de données,

- on part d'un domaine, dénoté par dom, qui est infini ;
- on requiert que toutes les relations soient finies ;
- on se limite aux formules qui sont *d☺main-independent* et, par conséquence, il est obsolète de connaître dom. Même l'observation "dom est infini" devient sans importance.

Ce choix correspond à la pratique : ce serait contre-intuitif que la réponse à une requête dépende des valeurs qui ne sont pas présentes dans les tables.

Non-existence d'un algorithme pour tester si une formule est *d☺main-independent*

Prouvez :

$\exists x\,(\neg S(x)) \vee (\varphi_1 \leftrightarrow \varphi_2)$ *est* *d☺main-independent* *ssi* $\varphi_1 \equiv \varphi_2$.

Preuve :

- Si $\varphi_1 \equiv \varphi_2$, cette formule est vraie par rapport à toute structure relationnelle $\Longrightarrow$ *d☺main-independent* ;
- si $\varphi_1 \not\equiv \varphi_2$, il existe une structure $(\mathcal{D}, \mathcal{I})$ telle que $(\mathcal{D}, \mathcal{I}) \not\models \varphi_1 \leftrightarrow \varphi_2$. Par rapport à cette structure, la formule est vraie si $\exists x\,(\neg S(x))$ est vraie $\Longrightarrow$ *d☹main-dependent*.

Variables libres

- La variable x a une occurrence libre dans la formule $\exists y\,(\mathit{Fleur}(x, y))$.
- La vérité d'une formule par rapport à une structure est seulement définie pour des formules sans variables libres.
- Une formule avec variable libre x définit une fonction (ou requête) : par rapport à une structure $(\mathcal{D}, \mathcal{I})$, renvoyer toute valeur $c \in \mathcal{D}$ telle que la formule est vraie si l'on remplace x par c.
- $\exists y\,(\mathit{Fleur}(x, y)) \rightsquigarrow$ renvoyer les espèces de fleurs ;
  $\mathit{Fleur}(x, y) \rightsquigarrow$ renvoyer les fleurs avec leurs couleurs.
- On utilisera la notation suivante :

$$\{x \mid \exists y\,(\mathit{Fleur}(x, y))\}$$
$$\{x, y \mid \mathit{Fleur}(x, y)\}$$

Requêtes peuvent être *d☹main-dependent* (et donc incorrectes)

$$\{x \mid \neg \exists y \, (Fleur(x, y))\}$$

$$\{x \mid \exists y \, (Fleur(x, y)) \vee \exists u \, (Couleur(u))\}$$

Examples

VINS

Cru	*Millesime*	*Qualite*
Chablis	1992	excellent
Chablis	1993	bon
Rothschild	1993	bon
Rothschild	1994	imbuvable

ABUS

Nom	*Cru*	*Annee*
Ed	Chablis	1992
Ed	Chablis	1993
An	Chablis	1993
An	Rothschild	1993

Qui n'a jamais bu un vin excellent ?

$$\{x \mid \overbrace{\exists u \exists m\,(ABUS(x,u,m))}^{\pi_{\{Nom\}}(ABUS)} \wedge \neg \underbrace{\exists u \exists m\,(ABUS(x,u,m) \wedge VINS(u,m,\text{'excellent'}))}_{\pi_{\{Nom\}}(E_1 \bowtie E_2)}\}$$

Comparaison avec l'algèbre SPJRUD. Soit

$$\begin{aligned} E_1 &= \rho_{Annee \mapsto Millesime}(ABUS) \\ E_2 &= \sigma_{Qualite = \text{“excellent”}}(VINS) \\ E_3 &= \pi_{\{Nom\}}(E_1 \bowtie E_2) \end{aligned}$$

E_3 renvoie les personnes ayant déjà bu un vin excellent. La requête demandée est donc :

$$\pi_{\{Nom\}}(ABUS) - E_3$$

Quels crus ont varié en qualité ?

$$\left\{ u \mid \exists q \exists v \left(\begin{array}{l} \underbrace{\exists m\,(VINS(u,m,q)) \wedge \exists m\,(VINS(u,m,v))}_{E_2(u,q,v)} \\ \wedge \neg\,(q = v) \end{array} \right) \right\}$$

Soit

$$\begin{array}{rcl} E_1 & = & \pi_{\{Cru,\,Qualite\}}(VINS) \\ E_2 & = & E_1 \bowtie \rho_{Qualite \mapsto Valeur}(E_1) \end{array}$$

E_2 renvoie $\{Cru : c, Qualite : q, Valeur : v\}$ ssi c est un cru qui a été de qualité q et de qualité v (il est possible que $q = v$). Soit

$$E_3 = E_2 - \sigma_{Qualite=Valeur}(E_2)$$

E_3 renvoie $\{Cru : c, Qualite : q, Valeur : v\}$ ssi c est un cru qui a été de qualité q et de qualité v avec $q \neq v$. La requête demandée est donc :

$$\pi_{\{Cru\}}(E_3)$$

Qui a bu tous les crus ?

$$\left\{ x \mid \begin{array}{c} \exists u \exists m\,(ABUS(x,u,m)) \\ \wedge\, \forall u \forall m \forall q\,(VINS(u,m,q) \rightarrow \exists m\,(ABUS(x,u,m))) \end{array} \right\}$$

$$\left\{ x \mid \begin{array}{c} \exists u \exists m\,(ABUS(x,u,m)) \\ \wedge\, \neg \exists u\,((\exists m \exists q VINS(u,m,q)) \wedge \neg \exists m\,(ABUS(x,u,m))) \end{array} \right\}$$

Soit

$$\begin{aligned} E_1 &= \pi_{\{Nom\}}(ABUS) \bowtie \pi_{\{Cru\}}(VINS) \\ E_2 &= \pi_{\{Nom,Cru\}}(ABUS) \\ E_3 &= E_1 - E_2 \end{aligned}$$

E_3 renvoie $\{Nom : n, Cru : c\}$ ssi c est un cru qui n'a pas été bu par la personne n. La requête demandée est donc :

$$\pi_{\{Nom\}}(ABUS) - \pi_{\{Nom\}}(E_3)$$