Langages d'interrogation relationnels

Université de Nice Sophia-Antipolis

Version 3.5.1 – 21/10/06

Richard Grin

Schéma relationnel utilisé pour ce cours

Employé(matr, nomE, poste, dateEmb, *sup*, salaire, commission, *dept*)

Dept(dept, nomD, lieu)

Projet(codeP, nomP)

Participation(*matr*, *codeP*, fonctionP)

Matricule du chef

Les clés primaires sont soulignées.
Les clés étrangères sont en italiques.

Exemples de tuples

- **Dept**(*dept*, nomD, lieu)
  (10, ‘Finances’, ‘Nice’)
- **Employé**(*matr*, nomE, poste, dateEmb, sup, salaire, commission, dept) :
  (109, ‘Dupond’, ‘Ingénieur’, ’10/10/2003’, 210, 1800, 0, 10)
- **Projet**(*codeP*, nomP)
  (‘P07’, ‘Qualité’)
- **Participation**(*matr*, *codeP*, fonctionP)
  (109, ‘P07’, ‘Chef’)

Langages d'interrogation relationnels

Proposition

- Une proposition est une déclaration qui est sans équivoque vraie ou fausse
- 2 + 2 = 4
- 2 + 2 = 5
- Dupond est un employé du département 10

Prédicats

- Un prédicat P est une phrase qui peut comporter des paramètres et qui peut être vraie ou fausse :
  P(x, y) = « x est un employé du département y »

Calcul des prédicats

- La logique du 1er ordre, ou calcul des prédicats, est la théorie mathématique qui étudie les formules logiques formelles (sans signification particulière) que l'on peut former avec des prédicats
- Elle permet de se donner des formules logiques et d'en déduire de nouvelles formules

Calcul des prédicats

- Les formules logiques que l'on étudie sont construites avec
  - des prédicats (P, Q,…)
  - des connecteurs logiques (et, ou, non, $\Rightarrow$)
  - des quantificateurs $\forall$ et $\exists$
  - des variables (x, y,…)
  - des constantes (a, b,…)
  - des fonctions (f, g,…)

Exemples de formules logiques

- P(x,y) et Q(x,a)
- non(P(x,y) et Q(x,a))
- $\forall$ x $\exists$ y (non(P(x,y) et Q(x,a)))
- P(x, f(a)) $\Rightarrow$ $\forall$ z $\exists$ y (non(P(z,y) et Q(x,a)))
- (P $\Rightarrow$ Q) $\Leftrightarrow$ ((non P) ou Q)
  (A$\Leftrightarrow$B signifie (A $\Rightarrow$ B) et (B $\Rightarrow$ A)) ; cette formule est toujours vraie

Sémantique

- Les domaines du discours sont les domaines dans lesquels les constantes et variables prennent leurs valeurs

Données décrites par une formule

- On peut choisir comme domaines du discours des domaines liés à une base de données relationnelle
- À toute formule du calcul des prédicats correspond l'***ensemble des valeurs des variables qui rendent vraie cette formule***
- Une formule du calcul des prédicats correspond ainsi à un sous-ensemble des données d'une BD

Exemple

- Domaine du discours = les employés et les départements d'une entreprise
- Prédicats :
  TRAVAILLE(e, d) est vrai si l'employé e travaille dans le département d
  DIRIGE(e1, e2) est vrai si e1 est chef de e2
- La formule TRAVAILLE(e, 'Comptabilité') correspond aux employés du département Comptabilité

Variantes du calcul relationnel

- 2 variantes pour décrire des sous-ensembles d'une base de données :
  – calcul relationnel des tuples
  – calcul relationnel des domaines

Calcul relationnel des tuples

- Fondé sur le calcul des prédicats, avec les variables qui prennent leur valeur dans l'**ensemble des tuples**
- Noms des employés qui participent à un projet :

Ce prédicat signifie « **E** est un tuple de **EMP** »

```
{E.nomE / EMP(E) ∧
          ∃ P PARTICIPATION(P) ∧
          (P.matr = E.matr)}
```

(les variables sont **E** et **P** ; ce sont des tuples)

E.nomE = **nomE** de **E**

Calcul relationnel des domaines

- Fondé sur le calcul des prédicats, avec les variables qui prennent leur valeur dans l'**ensemble des domaines**
- A inspiré le langage QBE (*Query By Example*)
- Noms des employés qui participent à un projet :

```
{ y / ∃ x EMP(matr:x, nomE:y) ∧
       PARTICIPATION(matr:x) }
```

(les variables sont **x** et **y** ; **x** prend ses valeurs dans le domaine de l'attribut **matr**)

« un tuple de **PARTICIPATION** a la valeur **x** pour son attribut **matr** »

Algèbre relationnelle

- Le principe est différent pour ce langage d'interrogation qui a inspiré le langage SQL
- Ce langage n'est pas descriptif mais « opératoire »
- Pour obtenir les données que l'on cherche, on donne la suite d'opérations à appliquer aux relations de la base
- 2 types d'opérateurs : relationnels et ensemblistes

Avertissement

- Les notations des opérateurs qui vont suivre sont des notations « maison »
- Raisons : les notations ne sont pas vraiment normalisées et/ou comportent des caractères « exotiques » non disponibles dans les polices de caractères standard

Opérateurs relationnels portant sur 1 relation

- Soit une relation **R**(A_1, …, A_n)
- Projection sur les attributs **Aj, …, Ak**, notée **R**[A_j, …, A_k]
  Exemple :
  Employé[matricule, nom, salaire]
- Sélection des t-uples qui vérifient une condition c, notée **R / c**
  Exemple :
  Employé / salaire > 1500

Opérateurs relationnels portant sur 2 relations

- Jointure (équi-jointure) de 2 relations R et S sur 2 attributs B et C, notée
  `R J{B=C} S`
- Division de 2 relations sur l'attribut B, notée
  R $\div_B$ S

Équi-jointure

- Opérateur fondamental de la théorie des bases de données relationnelles
- Il permet de créer une relation qui comprend des données venant de 2 relations
- Soient R(A, B) et S(C, D), B et C 2 attributs de types compatibles,
  la jointure **`R J{B=C} S`** est la relation dont les tuples sont obtenus par concaténation des tuples de R et de S qui ont la même valeur pour les attributs B et C

Exemple d'équi-jointure

Employé

matricule	**nom**	**dept**
1050	Dupond	10
832	Durand	20
900	Duval	10

Département

dept	**nom**	**lieu**
10	Finances	Paris
20	Ventes	Nice

Notation simplifiée :
Employé J{dept} Département

quand le nom des 2 colonnes est le même

Employé J{dept=dept} Département

matricule	**nom**	**dept**	**dept**	**nom**	**lieu**
1050	Dupond	10	10	Finances	Paris
832	Durand	20	20	Ventes	Nice
900	Duval	10	10	Finances	Paris

Exemple de jointure *naturelle*

Employé

matricule	**nom**	**dept**
1050	Dupond	10
832	Durand	20
900	Duval	10

Département

dept	**nom**	**lieu**
10	Finances	Paris
20	Ventes	Nice

Employé $\mathbf{J_N}${dept=dept} Département

matricule	**nom**	**dept**	**nom**	**lieu**
1050	Dupond	10	Finances	Paris
832	Durand	20	Ventes	Nice
900	Duval	10	Finances	Paris

La colonne de jointure n'est pas répétée

Jointure en général

- On peut remplacer = par un opérateur de comparaison >, <, ≥, ≤, ≠
- La jointure est alors notée, par exemple,
  R J{C < D} S
- Exemple :
  Employé J{salaire < salaire} Employé
- Pour lever les ambiguïtés :
  Employé e1 J{e1.salaire < e2.salaire} Employé e2

Division

R

A	**B**
x	1
y	2
z	1
x	3

S

C
1
3

R $\div_B$ S

A
x

Division euclidienne :
le quotient q est le *plus grand entier* tel que
b × q <= a

- **R $\div_B$ S** = les A de R qui sont associés à tous les C
  = {a ∈ R[A] / ∀c ∈ S, (a, c) ∈ R}
  = {a ∈ R[A] / $\boxed{\nexists}$ c ∈ S, (a, c) $\boxed{\notin}$ R}
- "Division", car c'est le plus grand sous-ensemble D de R[A] tel que D × S est inclus dans R

Quand utilise-t-on la division ?

- La division fournit la réponse au type de question suivante : quels sont les A qui sont associés à **tous** les C ?

Exemple

- Matricules des employés qui participent à tous les projets
- Participation[matr, codeP] $\div_{\mathbf{codeP}}$ Projet[codeP]

Participation[matr, codeP]

Matr	codeP
25	P1
32	P1
25	P2
40	P2

Projet[codeP]

codeP
P1
P2

Exemple

- Noms de tous les employés qui gagnent moins de 1500 euros
- (Employé/salaire < 1500)[nomE]
- ~~Mais pas (Employé[nomE])/salaire < 1500~~
- Autre présentation :
  R1 = Employé/salaire < 1500
  R = R1[nomE]

Opérateurs ensemblistes

- Pour 2 relations qui ont des attributs de types compatibles :
  - Réunion : R1 $\cup$ R2
  - Intersection : R1 $\cap$ R2
  - Différence : R1 **-** R2
- Pour 2 relations quelconques,
  - Produit cartésien : R1 $\times$ R2

Équivalence des langages

- On peut démontrer que
  - le calcul relationnels des t-uples,
  - le calcul relationnel des domaines,
  - l'algèbre relationnelle

  sont équivalents : ils permettent de décrire les mêmes ensembles de données dans une base de données

Un ensemble dans les 3 langages

Noms des employés qui participent à un projet :

```
1. E = {E.nomE / EMP(E) ∧
                  ∃ P PARTICIPATION(P)
                  ∧ (P.matr = E.matr)}
2. E = { y / ∃ x EMP(matr:x, nomE:y)
              ∧ PARTICIPATION(matr:x) }
3. R1 = EMP J{matr} PARTICIPATION
   E = R1[nomE]
```