LANGAGES RELATIONNELS

Des langages de définition et de manipulation sont associés au modèle relationnel. De façon formelle on trouve deux classes de langages. Les langages algébriques et les langages prédicatifs. Ces langages sont strictement équivalents sur le plan de la puissance d'expression (n'importe quelle requête qui peut s'exprimer dans un de ces langages peut s'exprimer dans l'autre). Sur un plan industriel, les langages algébriques ont dérivé SQL (Structured Query Language, défini par IBM et qui s'impose comme la norme supportée par tout SGBD du commerce) et les langages prédicatifs QBE (Query By Example).

Table des matières

I. L'algèbre relationnelle

L'algèbre relationnelle se compose d'un ensemble d'opérateurs opérant sur des relations et produisant de nouvelles relations. Il est donc possible de construire de nouvelles informations à partir des relations de départ et d'une composition séquentielle d'opérateurs.

De nombreux opérateurs relationnels ont été proposés, on peut cependant présenter ici les plus courants. On peut classifier les opérateurs relationnels en trois catégories :

- les opérateurs unaires : affectation, sélection et projection
- les opérateurs binaires travaillant sur des relations de même schéma : union, intersection, différence
- les opérateurs binaires travaillant sur des relations de schémas différents : jointure, produit cartésien, théta-jointure, division

La présentation des opérateurs est illustrée par un exemple de gestion d'une base d'invitations. Cette base de données décrit les personnes invitées et les plats qui ont été servis. Elle est composée de trois relations :

- REPAS(date,invité) donne la liste des invités qui ont été reçus et à quelle date
- MENU(date,plat) donne le menu servi à chaque date
- PREFERENCE(personne,plat) donne pour chaque personne ses plats préférés

N.B : les attributs "personne" et "invité" ont même domaine et les clés sont en soulignéees.

I.1 Opérateurs unaires

- **affectation :** R(A1, ..., An) <-- expression de sélection

  L'affectation permet de sauvegarder le résultat d'une expression de recherche, ou bien de renommer une relation et ses attributs.

Exemple :

Q1 : Quels sont les invités du repas du 010597.

en algèbre :

R1 <- SELECTION date=010597 (REPAS)
PROJECTION invité (R1)
ou bien sous forme fonctionnelle :
PROJECTION invité (SELECTION date=010597 (REPAS))

en SQL :

```
SELECT distinct invité FROM REPAS
WHERE date='010597'
```

en calcul nuplet :

{r[invité] / REPAS(r) ET r[date]=010597}

en calcul domaine :

{i / REPAS(i, 010597)}

- **sélection** : SELECTION condition-de-sélection (R)

  La sélection prend en entrée une relation R définie sur un schéma SR et produit en sortie une nouvelle relation de même schéma SR ayant comme nuplets ceux de R satisfaisant à l'expression de sélection. Une expresion de sélection est une condition booléenne construite à partir des connecteurs logiques et, ou, non et de conditions simples (attribut de R - opérateur de comparaison - constante ou attribut de R - opérateur de comparaison - attribut de R)

  Exemples :

Quels sont les plats qui ont été servis à Alice ?

en algèbre :

PROJECTION plat (SELECTION invité=Alice(REPAS) * MENU)

en SQL :

```
SELECT distinct plat
FROM REPAS R, MENU M
```

WHERE R.date=M.date AND invité='Alice'

en calcul nuplet :

{m[plat] / ILEXISTE r (MENU(m) ET REPAS(r) ET m[date]=r[date] ET r[invité]=Alice)}

en calcul domaine :

{p / ILEXISTE d (MENU(d, p) ET REPAS(d, Alice))}

la "jointure" est représentée par le fait d'utiliser la même variable d dans MENU et REPAS.

- **projection** : PROJECTION A1, ..., An (R)

La projection prend en entrée une relation R définie sur un schéma SR et produit en sortie une nouvelle relation de schéma A1, ..., An (schéma inclus dans SR) ayant comme nuplets ceux de R restreints au sous-schéma A1, ..., An. Il faut noter que la cardinalité de la nouvelle relation est inférieure ou égale à celle de R, puisque des doublons ont pu être produits par la projection et sont donc supprimés (une relation est toujours un ensemble).

Exemples : Q3, Q5, Q6

Q3 : Quels sont les invités qui lors d'un repas ont eu au moins un de leur plat préféré

en algèbre :

PREFERENCES1(personne1, plat1) <- PREFERENCES
PREFERENCES est renommée pour pouvoir faire une theta-jointure avec MENU (sinon un attribut est commun et la theta-jointure est alors impossible).
PROJECTION invité ((MENU * REPAS)*(invité=personne1 ET plat=plat1) PREFERENCES1)

en SQL :

```
SELECT distinct invité
FROM REPAS R, PREFERENCES P, MENU m
WHERE R.invité=P.personne AND
R.date=M.date AND P.plat=M.plat
```

en calcul nuplet :

{r[invité] / ILEXISTE m ILEXISTE p (REPAS(r) ET MENU(m) ET PREFERENCES(p) ET r[date]=m[date] ET m[plat]=p[plat] ET r[invité]=p[personne])}

en calcul domaine :

{i / ILEXISTE d ILEXISTE pl (REPAS(i, d) ET MENU(d, pl) ET PREFERENCES(i, pl))}

Q5 : Quelles sont les personnes qui n'ont jamais été invités ?

en algèbre :

PROJECTION personne (PREFERENCES) - PROJECTION invité (REPAS)

en SQL :

```
SELECT personne FROM PREFERENCES
EXCEPT (MINUS avec Oracle)
SELECT invité FROM REPAS
```

en calcul nuplet :

{p[personne] / QQSOIT r (PREFERENCES(p) ET REPAS(r) ET r[invité] != p[personne]) }

en calcul domaine :

{pe / ILEXISTE pl ILEXISTE d (PREFERENCES(pe, pl) ET REPAS(pe,d))}

Q6 : Quels sont les invités qui sont venus à tous les repas ?

en algèbre :

REPAS / PROJECTION date (REPAS)

en SQL :

```
SELECT invité FROM REPAS
GROUP BY invité
HAVING count(*) =
    (SELECT count(distinct date)
    FROM REPAS)
```

en calcul nuplet :

EDATE <- {r[date] / REPAS(r)}
{r[invité] / QQSOIT e (REPAS(r) ET EDATE(e) ET r[date]=e[date])}

en calcul domaine :

{i / QQSOIT d (REPAS(i, d))}

I.2 Opérateurs binaires de même schéma

Les trois opérateurs ensemblistes opèrent sur des relations R et S de même schéma SRS.

- **union** : R UNION S

  produit une nouvelle relation de schéma SRS ayant les nuplets de R et ceux de S (les doublons sont supprimés).

- **intersection** : R INTERSECTION S

produit une nouvelle relation de schéma SRS ayant les nuplets présents dans R et dans S.

- **différence** : R - S

  produit une nouvelle relation de schéma SRS ayant les nuplets de R qui ne sont pas dans S. Attention, la différence n'est pas commutative.

Exemple : Q5 (ci-dessus)

I.3 Opérateurs binaires de schémas différents

Soient R et S deux relations définies sur les schémas SR et SS.

- **jointure** : R * S

  Il faut que les schémas SR et SS aient une intersection SRS non vide. La relation produite a un schéma qui est l'union des schémas SR et SS et dont les nuplets sont la concaténation des nuplets de R avec ceux de S s'ils ont même valeur pour tous les attributs communs (c'est à dire ceux de SRS). On parle également de jointure naturelle ou équi-jointure.
  Exemple : Q2 (ci-dessus)

- **produit cartésien** : R X S

  Il faut que les deux schémas SR et SS soient disjoints. La relation produite a un schéma qui est l'union des schémas SR et SS et dont les nuplets sont la concaténation des nuplets de R avec ceux de S. Attention le nombre d'éléments du produit cartésien est le produit des cardinalités des relations R et S!

Exemple :

Q4 : Quel est le produit cartésien entre REPAS et PREFERENCES ?
en algèbre :

REPAS X PREFERENCES

en SQL :

```
SELECT R.*, P.*
FROM REPAS R, PREFERENCES P
```

en calcul nuplet :

{r, p / REPAS(r) ET PREFERENCES(p)}

en calcul domaine :

{d, i, pe, pl / REPAS(d, i) ET PREFERENCES(pe, pl)}

- **théta-jointure** : R * (condition) S

  Cette opération non canonique est équivalente à un produit cartésien suivi

d'une opération de sélection.
R * (condition) S = SELECTION condition (R X S)

Exemple : Q3 (ci-dessus)

- **division** : R / S

R est définie sur un schéma SR composé des ensembles d'attributs X,Y (SR = X UNION Y) et S est définie sur un schéma SS composé de l'ensemble d'attributs Y (SS = Y). La relation produite est définie sur le schéma Sres (Sres = SR - SS = Y) et comprend tous les nuplets dont la concaténation avec tous les nuplets de S appartient à R (résultat X S inclus-ou-egal R).
La division peut se définir à l'aide du produit cartésien, de la différence et de la projection :

R(A1, .., Ap, Ap+1, ..., An)
S(Ap+1, ..., An)
R / S = R1 - R2 avec
R1 = PROJECTION A1, ..., Ap (R)
R2 = PROJECTION A1, ..., Ap ((R1 X S) - R)

Exemple : Q6

Un des grands intérets de l'algebre relationnelle est l'ensemble de propriétés des opérateurs ce qui permet l'optimisation des requetes.

I.4 Règles de passage de l'algèbre relationnelle vers SQL

Soient R(X) et S(Y) deux relations de schémas distincts, T(V,W), U(W,Z) deux relations ayant un ensemble d'attributs communs, Q1(X), Q2(X) deux relations de même schéma et P(W).

sélection	SELECTION condition (R)	select *from R where condition
projection	PROJECTION A1, ..., An (R)	select distinct A1, ..., An from R
opérateurs ensemblistes	Q1 UNION Q2, Q1 Q1 INTERSECTION Q2 Q1 - Q2	select * from Q1 union, intersect, minus select * from Q2
jointure	T * U	select V, W, Z from T, U where T.W=U.W (en réalité égalité sur tous les attributs communs)
produit cartésien	R X S	select R.*, S.* from R, S
théta-jointure	R* (condition) S	select X, Y from R, S

		where condition
division	T / P avec P partie de T	select V from T group by V having count(distinct W) >= (select count(distinct W) from P)

Exemple complet de la base des invitations

II. Les langages prédicatifs (nuplet et domaine)

Deux approches sont possibles pour établir un lien entre la logique du premier ordre et les bases de données :

II.1 Approche "intelligence artificielle" : BD déductives

Les faits élémentaires (données) et les propriétés générales (schéma ainsi que les règles d'intégrité et d'inférences) sont représentés par des axiomes d'une théorie du premier ordre. Répondre à une question revient donc à démontrer un théorême dans cette théorie.

Avantages : une question est résolue en dehors de toute interprétation et on peut utiliser des règles d'intégrité ou d'inférences (qui permettent de déduire des informations non stockées).

Inconvénients : problème de performances (il faut intégrer un démonstrateur de théorêmes) et représentation de l'information négative. La plupart du temps on utilise l'hypothèse du monde fermé ("Closed World Assumption") avec une méta-règle de négation par l'échec ("negation as failure"). Toute information manquante est considérée comme fausse et une information qu'on ne peut dériver est également considérée comme fausse.

II.2 Approche "classique" : BD relationnelle

Les propriétés générales (schéma seulement) sont considérées comme les axiomes d'une théorie du premier ordre, alors que les faits élémentaires (données) sont perçus comme une interprétation de cette théorie. Si cette interprétation vérifie tous les axiomes, elle constitue un modèle. Répondre à une question revient à chercher la valeur de vérité d'une formule relativement à l'interprétation. On distingue alors entre question fermée (sans variable libre) pour laquelle la réponse est vrai ou faux, et question ouverte (avec variable libre) pour laquelle la réponse sont les valeurs de l'interprétation pour lesquelles la question est vraie.

Avantages : on retrouve la théorie relationnelle classique

Inconvénients : pas de puissance supplémentaire

Nous présentons dans la suite cette deuxième approche, les BD déductives étant étudiées dans le cadre du cours de 3ème année. On rencontre deux classes de langages prédicatifs, ceux à variable nuplet (le domaine d'interprétation d'une variable est le nuplet) et ceux à variable domaine (le domaine d'interprétation d'une variable est le domaine).

II.3 Spécification formelle du calcul relationnel à variable nuplet

Une expression générale du calcul relationnel à variable nuplet est de la forme :

$$\{t1[A1], t2[A2], ..., tn[An] / COND(t1, t2, ..., tn, tn+1, tn+2, ..., tn+m)\}$$

avec ti (de 1 à n+m) variables nuplets non nécessairement distinctes deux à deux, Ai sont des attributs de relations associées aux ti et COND est une formule du calcul relationnel à variable nuplet.

L'expression de la forme ti[Aj] exprime la valeur du nuplet ti suivant l'attribut de nom Aj (terme de projection).

Une formule est construite à partir de prédicats atomiques (atomes) qui sont d'une des formes suivantes :

1.Un atome de la forme R(ti) où R est un nom de relation et ti est une A variable nuplet. Cet atome identifie le domaine de la variable ti comme celui de la relation de nom R.

2.Un atome de la forme ti[A] op tj[B] où op est un opérateur de comparaison classique, ti et tj des variables nuplets, et A, B des attributs des relations associées.

3.Un atome de la forme ti[A] op c ou c op tj[B], où c est une constante.

Une formule est constituée d'atomes connectés par les opérateurs logiques ET, OU, NON et se définit de la manière suivante :

1.Tout atome est une formule,

2.Si F1 et F2 sont des formules, alors (F1 et F2), (F1 ou F2), non(F1) sont des formules,

3.Si F est une formule, alors (ILEXISTE t) (F) est une formule avec t variable nuplet,

4.Si F est une formule, alors (QQSOIT t) (F) est une formule avec t variable nuplet.

NB : une variable est dite libre si elle n'est pas quantifiée dans sa portée, liée sinon.

II.4 Spécification formelle du calcul relationnel à variable domaine

Une expression générale du calcul relationnel à variable domaine est de la forme :

{x1, x2, ..., xn / COND(x1, x2, ..., xn, xn+1, xn+2, ..., xn+m)}

avec xi variables domaines (non nécessairement distinctes) d'attributs et COND est une formule du calcul relationnel à variable domaine.

Une formule est construite à partir d'atomes. Un atome peut être d'une des formes suivantes :

1.Un atome de la forme R(x1, x2, ..., xj) où R est le nom d'une relation de degré j et chaque xi est une variable. <x1, ..., xj> représente un nuplet de la relation R et xi la ième valeur dans le nuplet.

2.Un atome de la forme xi op xj où op est un opérateur de comparaison classique et xi, xj des variables domaines.

3.Un atome de la forme xi op c ou c op xj, où c est une constante.

Une formule est construite à partir des mêmes règles que celles définies pour le calcul relationnel à variable nuplet.