

Introduction à l'algèbre relationnelle

Table des matières

1	Introduction	1
1.1	Origine	1
1.2	Rappels terminologiques	2
1.3	Exemples de relations	7
1.4	Intégrité des relations	9
1.5	Définitions des schémas de relations	11
1.6	Algèbre relationnelle	11
2	Opérateurs unaires	14
2.1	Projection	14
2.2	Sélection	16
2.3	Renommage	18
3	Opérations ensemblistes	19
3.1	Rappels	19
3.2	Union	20
3.3	Différence	22
3.4	Produit cartésien	24
4	Opérations dérivées	27
4.1	Intersection	27
4.2	Jointure interne	29
4.3	Jointure externe	33
4.4	Cas particuliers de jointures	35
4.5	Division	37
5	Extensions	41
5.1	Agrégats de n-uplets	41
5.2	Calculs de valeurs d'attributs	43
5.3	Classement des n-uplets	43
5.4	Opérations de mise à jour	44

6	Expressions relationnelles	45
6.1	Composition des opérations	45
6.2	Équivalences	45
6.3	Représentation arborescente	46
7	Estimation du cout d'une expression relationnelle	47
7.1	Selection	48
7.2	Projection	48
7.3	Produit cartésien	48
7.4	Thétajointure	48
7.5	Jointure naturelle	49
7.6	Exemple	49
8	Mémo - Opérateurs de l'algèbre relationnelle	51
9	Exercices	52
9.1	Questions	52
9.2	Propositions de réponses :	53

1 Introduction

1.1 Origine

Le modèle relationnel a été introduit dans les années 1970 par Edgar Frank Codd ("Ted" Codd, 1923-2003), alors informaticien chez IBM, comme moyen de définir et représenter les données de grandes bases de données d'une manière simple mais rigoureuse [Codd, 1969].

L'algèbre relationnelle complète ce modèle [Codd, 1970] en formalisant les opérations applicables aux relations afin d'en extraire des informations afin de répondre à des questions. Ces opérations deviendront le socle théorique des langages d'accès aux bases de données comme SQL¹.

S'en suivront des articles complétant la définition des règles, les formes normales, permettant de normaliser les relations afin de les rendre exemptes de défauts [Codd, 1971]).

¹Structured Query Language

1.2 Rappels terminologiques

Le modèle de Codd est le socle théorique de la grande majorité des bases de données de production à l'heure actuelle.

Cf. Article de EF Codd :

http://www.databaseanswers.org/downloads/Codds_1970_Paper.pdf

1.2 Rappels terminologiques

1.2.1 Ensemble

Ensemble

| Un ensemble est une collection non ordonnée d'objets distincts.

1.2.2 Domaine

Domaine

| Un **domaine** est un ensemble identifié de valeurs.

C'est la première forme de structuration des données d'un univers réel.

La notion de domaine est assimilable à celle de type de données (Codd : "semantical data type") utilisée en algorithmique ou programmation :

- nombres entiers : $[-\infty; \infty]$
- nombres entiers naturels : $[0; \infty]$
- nombres réels
- chaînes de caractères
- énumérations

Un domaine peut être plus précis :

- jours du mois : $[1; 31]$

- jours de la semaine : {dimanche, lundi, mardi, ..., vendredi, samedi}
- mois de l'année : {janvier, février, mars, ..., novembre, décembre}
- lettres minuscules de l'alphabet US : [a; z]
- couleurs des yeux : {bleu, vert, marron, noir}

1.2.3 Relation

La relation peut être vue comme une table de valeurs (ensemble de valeurs uniques et non ordonnées).

"The term relation is used here in its accepted mathematical sense. Given sets S_1, S_2, \dots, S_n , (not necessarily distinct), R is a relation on these n sets if it is a set of n -tuples each of which has its first element from S_1 , its second element from S_2 , and so on. (Note : More concisely, R is a subset of the Cartesian product $S_1 \times S_2 \times \dots \times S_n$) ..." ([Codd, 1970], 1.3 A Relational View of Data)

soit :

Le terme relation doit être entendu au sens mathématique. Soient les ensembles S_1, S_2, \dots, S_n , R est une relation sur ces n ensembles si c'est un ensemble de n -uplets (tuples à n éléments)², chacun ayant son premier élément dans S_1 , son second élément dans S_2 , et ainsi de suite. (Note : de manière plus concise, R est un sous-ensemble du produit cartésien $S_1 \times S_2 \times \dots \times S_n$)

Le terme n -uplets (tuples en anglais), qualifie l'ensemble de "couples" (si n vaut 2) formés par le produit cartésien de n ensembles. (voir figure 1 page 4)

² n -uplet : collection d'objets à n éléments ; un "2-uplet" est un couple, un "3-uplet" est un triplet, etc.

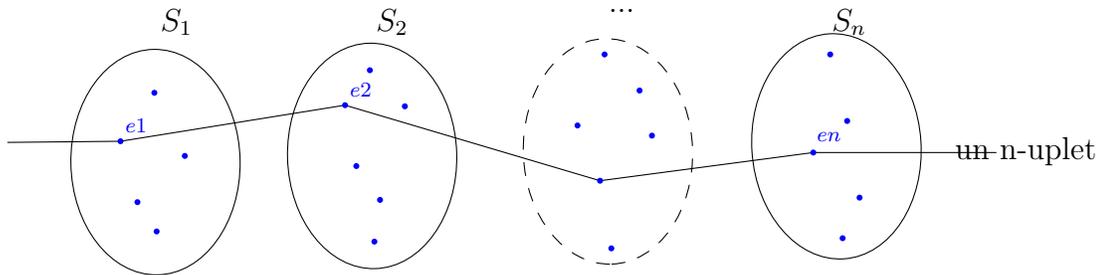


FIGURE 1 : Un n -uplet de la relation R , ses éléments e_1, e_2, \dots, e_n , appartiennent respectivement aux ensembles S_1, S_2, \dots, S_n .

Relation

Une **relation** est un sous-ensemble du produit cartésien d'une liste de domaines.

Rappel : Le produit cartésien de deux ensembles X et Y est l'ensemble de tous les couples, dont la première composante appartient à X et la seconde à Y .

Soit les domaines :

- jourMois : $[1; 31]$
- moisAnnee : {janvier, février, mars, ..., novembre, décembre}

$Anniversaire \subseteq jourMois \times moisAnnee$

Une relation $Anniversaire(jourMois, moisAnnee)$, qui contient la liste des dates pour lesquelles un anniversaire est à souhaiter, est bien un sous-ensemble du produit cartésien de l'ensemble des éléments de jourMois par l'ensemble des éléments de moisAnnee :

- le produit cartésien sera composé de toutes les combinaisons possibles de ces 2 ensembles
- mais la relation Anniversaire ne retiendra qu'un sous-ensemble de ce dernier.

1.2.3.1 Autres bases formelles à la notion de relation Une relation est également un prédicat à N variables : par exemple "l'anniversaire a lieu le jourmois du mois de moisAnnee" soit *Anniversaire(jourMois, moisAnnee)*

Une proposition devient "l'anniversaire a lieu le 16 du mois de mai".

Un tuple est une proposition VRAIE : exemple : *Anniversaire(16,"mai")*.

1.2.4 Attribut

Attribut

Un **attribut** est l'un des composants d'une relation. Chaque attribut est identifié et est associé à un domaine de valeur. Un même domaine peut être utilisé plusieurs fois dans une relation.

Par exemple, pour la relation 'Anniversaire', on peut définir les attributs suivants :

- jour appartenant au domaine jourMois
- mois appartenant au domaine moisAnnee

Un attribut doit avoir un nom unique dans une relation.

Seul le fait d'appartenir au même domaine permet aux attributs d'être comparés (et donc, des attributs qui ne sont pas du même domaine ne peuvent être comparés sémantiquement, même si leur type syntaxique est le même. Par exemple, un numéro de mois et une année sont tous deux des nombres entiers naturels, mais ils ne sont pas comparables car ils n'ont pas le même domaine, et donc pas le même sens (sémantique différente).

1.2.5 Schéma de relation

Schéma de relation

un schéma de relation définit l'ensemble des attributs et domaines d'une relation.

1.2 Rappels terminologiques

Le schéma de la relation devient ainsi :

Anniversaire (jour : jourMois, mois : moisAnnee)

Ainsi

- l'attribut jour est défini sur le domaine jourMois
- l'attribut mois est défini sur le domaine moisAnnee

1.2.6 Degré ou arité d'une relation

Degré

| Le degré d'une relation correspond à son nombre d'attributs.

Le degré de la relation Anniversaire est 2.

1.2.7 n-uplet, tuple

n-uplet

| Un n-uplet (en anglais : *tuple*) est une collection ordonnée de n éléments.

Il ne peut y avoir 2 n-uplets identiques dans une relation (pas de doublons).

Des n-uplets de la relation 'Anniversaire' pourraient être :

(16, mai), (29, juin), (6, octobre)

1.2.8 Cardinalité

Cardinalité

| La cardinalité de la relation correspond à son nombre de n-uplets.

1.3 Exemples de relations

La cardinalité de la relation Anniversaire est 3.

1.2.9 ...en résumé

Une manière courante de représenter les relations est la forme tabulaire (sous forme de tableau : cf. figure 2 page 7) :

- l'entête représente la définition de la relation,
- le corps représente la valeur de la relation (appelée *relval*).

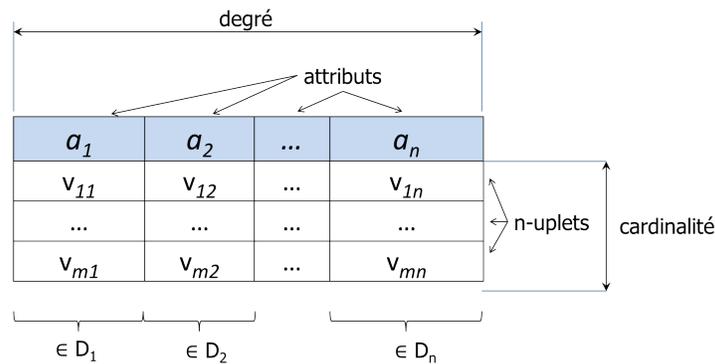


FIGURE 2 : une relation

1.3 Exemples de relations

Exemple : 1

Soit le domaine 'lettre' correspondant aux lettres de 'a' à 'z'.

Soit la relation R définie par les attributs A, B et C :

- en compréhension (ou en intention) :
 - $R(A : lettre, B : lettre, C : lettre)$
 - Degré de la relation : 3

1.3 Exemples de relations

- en extension :

$$R = ((a,b,c), (d,a,f), (c,b,d))$$

– Cardinalité de la relation 3

Ou bien sous forme tabulaire :

R	A	B	C
	a	b	c
	d	a	f
	c	b	d

Exemple : 2

On considère les relations "Etudiant", "Formation" et "Inscrire" :

- *Etudiant*(numero, nom, prenom, dateNaissance)

Etudiant	numero	nom	prénom	dateNaissance
	1	Dupont	Jacques	15/01/1990
	9	Durand	Pierre	20/07/1999
	23	Lambert	Paul	03/12/1994
	34	Durand	Jacques	15/12/1994
	105	Lambert	Paul	20/10/1995

- *Formation*(code, intitule)

Formation	code	intitule
	INFO	STS Informatique
	MATH	STS Mathematiques
	PC	STS Physique Chimie

- *Inscrire*(code, numero)

Inscrire	code	numero
	INFO	1
	INFO	9
	INFO	34
	MATH	1
	MATH	23
	MATH	34
	PC	1

1.4 Intégrité des relations

Mécanismes mis en oeuvre afin de contrôler l'intégrité des valeurs.

"...Normally, one domain (or combination of domains) of a given relation has values which uniquely identify each element (n-tuple) of that relation. Such a domain (or combination) is called a primary key...."

"...A common requirement is for elements of a relation to cross-reference other elements of the same relation or elements of a different relation. Keys provide a user-oriented means (but not the only means) of expressing such crossreferences. We shall call a domain (or domain combination) of relation R a foreign key if it is not the primary key of R but its elements are values of the primary key of some relation S..."

La notion de clef permet de maintenir l'intégrité des valeurs dans les relations :

- la *clé candidate* : attribut, ou ensemble minimal d'attributs, qui permet l'identification d'un n-uplet parmi tous les n-uplets d'une relation (un numéro, un code INSEE, une adresse mail)
 - on la représente généralement soulignée en pointillés : immatAv;
- la *clé primaire* (en anglais : *Primary Key*) : attribut, ou ensemble d'attributs, choisi parmi les clefs candidates, qui permet l'identification d'un n-uplet parmi tous les n-uplets d'une relation ; ³
 - on la représente généralement soulignée numP
- la *clé étrangère* (en anglais : *Foreign Key*) : un attribut (ou groupe d'attributs) clef étrangère dans une relation est clef primaire dans une autre relation ; elle matérialise un lien logique entre ces 2 relation ; ⁴

³l'intégrité d'entité sera le mécanisme mis en oeuvre dans les bases de données pour vérifier que la valeur d'une colonne clé primaire est unique dans une table et donne ainsi accès à une seule ligne de la table

⁴l'intégrité référentielle sera le mécanisme mis en oeuvre dans les bases de données pour vérifier que la valeur d'une colonne clé étrangère correspond à la valeur de la clé primaire de la table référencée

1.4 Intégrité des relations

- on la représente généralement préfixée, ou suffixée, du caractère # (dièse, AltGr+3 au clavier) : #*numAv* ou *numAv*#

Ainsi, 3 éléments garantissent l'intégrité d'une relation :

- domaine : intégrité de type
- clef primaire : intégrité d'entité
- clef étrangère : intégrité référentielle

Exemple : 1

- *Pilote*(*numP*, *villeP*)
- *Avion*(*numAv*, *nomAv*, *capaAv*, *locAv*, *immatAv*)
- *Vol*(*numVol*, *villeDep*, *villeArr*, #*numP*, #*numAv*)

avec :

- #*numP* de *Vol* fait référence à *numP* clef primaire de *Pilote*
- #*numAv* de *Vol* fait référence à *numAv* clef primaire de *Avion*

Exemple : 2

- *Ouvrage*(*num_ouvrage*, *titre*, *annee*, #*reliure*)
- *Reliure*(*reliure*, *prix*)

avec :

- #*reliure* de *Ouvrage* fait référence à *reliure* dans *Reliure*

1.5 Définitions des schémas de relations

Exemple : 3

- *Etudiant*(numero, nom, prenom, dateNaissance)
- *Formation*(code, intitule)
- *Inscrire*(#code, #numero)

avec :

- #numero de *Inscrire* fait référence à *numero* clef primaire de *Etudiant*
- #code de *Inscrire* fait référence à *code* clef primaire de *Formation*
- remarquer la clef primaire composée de *Inscrire*

1.5 Définitions des schémas de relations

1. définir les domaines dont certains seront "primaires" (= sur lesquels une clef primaire est définie)
2. définir les relations

La déclaration d'une relation équivaut à une déclaration de variables (prédicat : "relvar", soit un ensemble de paires <nom, domaine>).

La définition en extension correspond au corps, à la table des valeurs.

1.6 Algèbre relationnelle

Codd définit un langage de manipulation des relations : un langage ensembliste, l'algèbre relationnelle (ou algèbre de Codd) : on y décrit la manière d'obtenir une relation-résultat.

L'algèbre relationnelle est un langage :

- fermé : l'argument de sortie est du même type que l'argument d'entrée (on manipule des relations et on obtient une relation en résultat) ; cela permet la décomposition de questions complexes en questions simples

- complet : jeu d'opérateurs complet (tout nouvel opérateur pourra être basé sur les opérateurs de base) ;
- orthogonal : les opérateurs sont indépendants les uns des autres (pas de contraintes induites par un autre opérateur ni d'effet de bord)

L'algèbre relationnelle (ou langage algébrique de Codd) définit les opérations qui, appliquées à des relations, fournissent une nouvelle relation.

Elle comporte deux familles d'opérateurs de base :

- Les opérateurs unaires (mise en jeu d'une seule relation) :
 - projection : (en anglais : *project*), π (lettre grecque pi minuscule)
 - sélection : (en anglais : *select*), σ (lettre grecque sigma minuscule)
 - renommage : (en anglais : *rename*), ρ (lettre grecque rho minuscule)
- Les opérateurs ensemblistes (mise en jeu de 2 relations) :
 - union : (en anglais : *union*), \cup
 - différence : (en anglais : *set difference*), $-$
 - produit cartésien : (en anglais : *cartesian product*), X

Ces 6 opérations forment un ensemble de base cohérent et minimal.

Des extensions à ces opérations de bases permettent :

- d'en améliorer l'efficacité (simplification de leur écriture) :
 - intersection (ensembliste) : (en anglais : *intersect*), \cap
 - jointure interne : (en anglais : *inner join*), \bowtie
 - jointures externes : (en anglais : *outer join*), gauche, droite et complète, $\bowtie\leftarrow$, $\bowtie\rightarrow$, $\bowtie\lrcorner$
- d'effectuer des calculs sur des n-uplets en réalisant des agrégats
- d'effectuer des calculs sur des attributs en utilisant les opérateurs arithmétiques classiques

Les opérateurs ensemblistes nécessitent que les relations manipulées respectent la propriété d'uni(on)-compatibilité (même nombre d'attributs et même domaine pour chaque attribut pris 2 à 2).

Certains auteurs complètent ce jeu d'opérateurs :

- le classement des n-uplets d'une relation dans un certain ordre

Remarque : le symbolisme de représentation des opérateurs peut varier d'un auteur à l'autre. Vous trouvez les exemples de représentations courants.

L'algèbre relationnelle est un langage théorique d'interrogation des relations et forme la base des langages de requête appliqués aux bases de données relationnelles (SQL, par exemple).

2 Opérateurs unaires

Opérateurs unaires

Les opérateurs unaires (ou unitaires) s'appliquent à une seule relation et ont pour résultat une nouvelle relation.

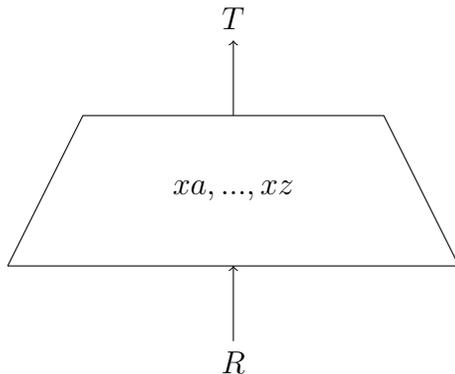
2.1 Projection

Projection

La projection est une opération de restriction sur les attributs.

La projection d'une relation R produit une nouvelle relation T en ne retenant que certains attributs. Afin de conserver la règle qui stipule que tous les n -uplets sont différents, les doublons sont supprimés du résultat.

2.1.1 Notation



$\pi_{x_a, \dots, x_z}(R)$ ou : PROJECT $R(x_a, \dots, x_z)$ R étant définie par les attributs x_1, \dots, x_n , la liste des attributs x_a, \dots, x_z formant un sous-ensemble de la liste d'attributs x_1, \dots, x_n .

2.1.2 Exemples

Exemple : 1

2.1 Projection

R	A	B	C
	a	b	c
	d	a	f
	c	b	d

$$T \leftarrow \pi_{a,b}(R)$$

T	A	B
	a	b
	d	a
	c	b

Exemple : 2

Etudiant	numero	nom	prénom	dateNaissance
	1	Dupont	Jacques	15/01/1990
	9	Durand	Pierre	20/07/1999
	23	Lambert	Paul	03/12/1994
	34	Durand	Jacques	15/12/1994
	105	Lambert	Paul	20/10/1995

$$T \leftarrow \pi_{nom,prenom}(Etudiant)$$

T	nom	prenom
	Dupont	Jacques
	Durand	Pierre
	Lambert	Paul
	Durand	Jacques

Le doublon de n-uplet (Lambert, Paul) a été supprimé.

$$T \leftarrow \pi_{prenom}(Etudiant)$$

T	prenom
	Jacques
	Pierre
	Paul

Le doublon de n-uplet (Paul) a été supprimé.

2.2 Sélection

Sélection

La sélection est une opération de restriction sur les n-uplets d'une relation.

La sélection consiste à extraire de la relation R les n-uplets satisfaisant à un prédicat (appelé aussi qualification Q ou plus simplement condition).

La qualification Q est exprimée à l'aide de

- d'opérandes : noms d'attributs et valeurs littérales
- d'opérateurs relationnels ($>$, $<$, $>=$, $<=$, $=$, $<>$)
- d'opérateurs spécifiques :
 - LIKE "modèle", où modèle contient un caractère joker : teste si une valeur correspond à un modèle (% remplace plusieurs caractères, _ remplace 1 caractère)
 - IN (liste de valeurs) : teste si une valeur se trouve dans une liste de valeurs
 - BETWEEN valeurMini AND valeurMaxi : teste si une valeur se trouve entre une valeur mini et une valeur maxi
- et de connecteurs logiques (\vee OR, \wedge AND , \neg NOT)

2.2.1 Notation



$$\sigma_Q(R) \text{ ou : SELECT R(Q)}$$

2.2.2 Exemples

Exemple : 1

R	A	B	C
	a	b	c
	d	a	f
	c	b	d

$$T \leftarrow \sigma_{B \neq 'a'}(R)$$

T	A	B	C
	a	b	c
	c	b	d

Exemple : 2

2.3 Renommage

Etudiant	numero	nom	prénom	dateNaissance
	1	Dupont	Jacques	15/01/1990
	9	Durand	Pierre	20/07/1999
	23	Lambert	Paul	03/12/1994
	34	Durand	Jacques	15/12/1994
	105	Lambert	Paul	20/10/1995

$$T \leftarrow \sigma_{numero < 10}(Etudiant)$$

T	numero	nom	prénom	dateNaissance
	1	Dupont	Jacques	15/01/1990
	9	Durand	Pierre	20/07/1999

2.3 Renommage

○ Renommage

L'opération de renommage est utilisée afin de rebaptiser des attributs afin de lever une ambiguïté lorsque, dans une opération binaire, un même identificateur est utilisé.

2.3.1 Notation

2.3.1.1 Renommer une relation $\rho_S(R)$

renomme la relation R en S

2.3.1.2 Renommer certains (ou tous) attributs d'une relation $\rho_{x1/nx1,x2/nx2,\dots,xn/n}$

retourne une nouvelle relation à partir du renommage de chacun des attributs de R : x_1, x_2, \dots, x_n en n_1, n_2, \dots, n_n , respectivement, $n_{x_1}, n_{x_2}, \dots, n_{x_n}$.

Les attributs qui ne sont pas dans la liste conserve leur nom.

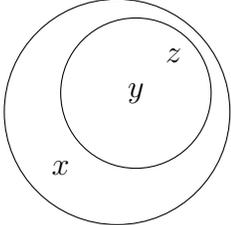
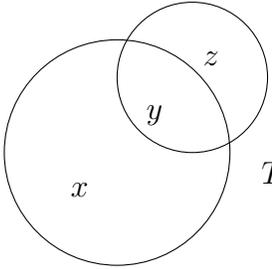
Le symbole ' \rightarrow ' a été privilégié à ' $/$ ' afin de pouvoir inclure des opérations de calcul

3 Opérations ensemblistes

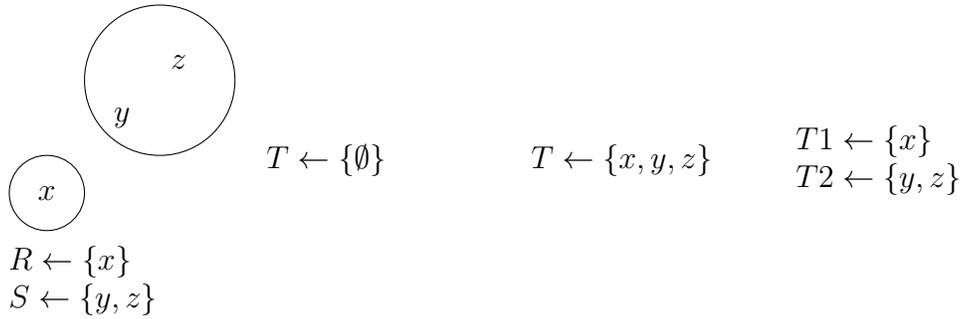
🔍 Opérateurs ensemblistes

Les opérateurs ensemblistes (ils sont binaires) s'appliquent à 2 relations et ont pour résultat une nouvelle relation.

3.1 Rappels

ensembles de départ	intersection $T \leftarrow R \cap S$	union $T \leftarrow R \cup S$	Différence $T1 \leftarrow R - S$ $T2 \leftarrow S - R$
 $R \leftarrow \{x, y, z\}$ $S \leftarrow \{y, z\}$	$T \leftarrow \{y, z\}$	$T \leftarrow \{x, y, z\}$	$T1 \leftarrow \{x\}$ $T2 \leftarrow \emptyset$
 $R \leftarrow \{x, y\}$ $S \leftarrow \{y, z\}$	$T \leftarrow \{y\}$	$T \leftarrow \{x, y, z\}$	$T1 \leftarrow \{x\}$ $T2 \leftarrow \{z\}$

3.2 Union



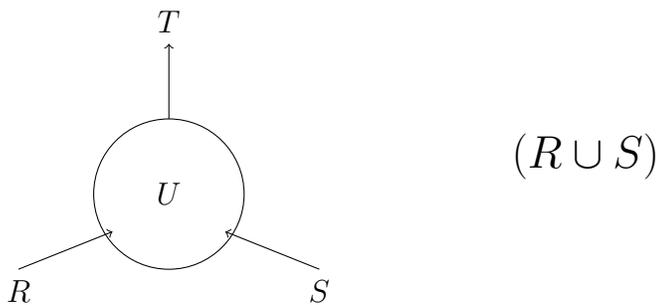
3.2 Union

Union

L'union de 2 relations R et S de même schéma est une relation T de même schéma contenant l'ensemble des n-uplets appartenant soit à R, soit à S, soit à la fois à R et S (les doublons sont supprimés), soit :
 $(R \cup S) = \{t : t \in R \vee t \in S\}$, opérateur logique OU.

Les relations R et S doivent être *union compatible* : même nombre d'attributs et même domaine pour chaque attribut.

3.2.1 Notation



L'union est commutative : $(R \cup S)$ est équivalent à $(S \cup R)$

3.2.2 Exemples

Exemple : 1

R	A	B	C
	a	b	c
	d	a	f
	c	b	d

-

S	A	B	C
	b	g	a
	d	a	f

R et S sont union-compatibles.

$$T \leftarrow (R \cup S)$$

T	A	B	C	
	a	b	c	provient de R
	d	a	f	provient de R et S : doublon supprimé
	c	b	d	provient de R
	b	g	a	provient de S

Exemple : 2

Etudiant	numero	nom	prénom	dateNaissance
	1	Dupont	Jacques	15/01/1990
	9	Durand	Pierre	20/07/1999
	23	Lambert	Paul	03/12/1994
	34	Durand	Jacques	15/12/1994
	105	Lambert	Paul	20/10/1995

$$R1 \leftarrow \sigma_{numero < 10}(Etudiant)$$

R1	numero	nom	prénom	dateNaissance
	1	Dupont	Jacques	15/01/1990
	9	Durand	Pierre	20/07/1999

$$R2 \leftarrow \sigma_{numero > 50}(Etudiant)$$

R2	numero	nom	prénom	dateNaissance
	105	Lambert	Paul	20/10/1995

$$T \leftarrow (\sigma_{numero < 10}Etudiant) \cup (\sigma_{numero > 50}Etudiant)$$

$$T \leftarrow (R1 \cup R2)$$

T	numero	nom	prénom	dateNaissance
	1	Dupont	Jacques	15/01/1990
	9	Durand	Pierre	20/07/1999
	105	Lambert	Paul	20/10/1995

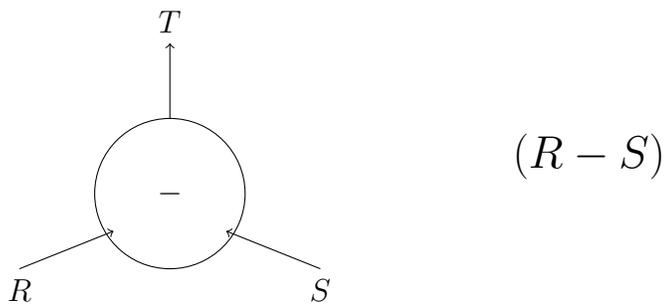
3.3 Différence

🔍 Différence

La différence entre 2 relations R et S de même schéma dans l'ordre (R - S) est la relation T de même schéma contenant les n-uplets appartenant à R sauf ceux qui appartiennent aussi à S, soit $(R - S) = \{t : t \in R \wedge t \notin S\}$, opérateur logique NON

Les relations R et S doivent être *union compatible* : même nombre d'attributs et même domaine pour chaque attribut.

3.3.1 Notation



La différence n'est pas commutative : $(R - S)$ n'est pas équivalent à $(S - R)$.

3.3.2 Exemples

Exemple : 1

3.3 Différence

R	A	B	C		S	A	B	C
	a	b	c	-		b	g	a
	d	a	f			d	a	f
	c	b	d					

$$T \leftarrow (R - S)$$

T	A	B	C
	a	b	c
	c	b	d

Exemple : 2

Etudiant	numero	nom	prénom	dateNaissance
	1	Dupont	Jacques	15/01/1990
	9	Durand	Pierre	20/07/1999
	23	Lambert	Paul	03/12/1994
	34	Durand	Jacques	15/12/1994
	105	Lambert	Paul	20/10/1995

$$R1 \leftarrow \sigma_{numero > 10}(Etudiant)$$

R1	numero	nom	prénom	dateNaissance
	23	Lambert	Paul	03/12/1994
	34	Durand	Jacques	15/12/1994
	105	Lambert	Paul	20/10/1995

$$R2 \leftarrow \sigma_{numero > 100}(Etudiant)$$

R2	numero	nom	prénom	dateNaissance
	105	Lambert	Paul	20/10/1995

$$T \leftarrow (\sigma_{numero > 10}Etudiant) - (\sigma_{numero > 100}Etudiant)$$

$$T \leftarrow (R1 - R2)$$

T	numero	nom	prénom	dateNaissance
	23	Lambert	Paul	03/12/1994
	34	Durand	Jacques	15/12/1994

3.4 Produit cartésien

🔍 Produit cartésien

Le produit cartésien de 2 relations R et S de schéma quelconque est une relation T ayant pour attributs les attributs de R et de S, et dont les n-uplets sont constitués par la combinaison de chaque n-uplet de R avec chacun des n-uplets de S, soit $(R \times S) = \{(r, s) : r \in R \wedge s \in S\}$.

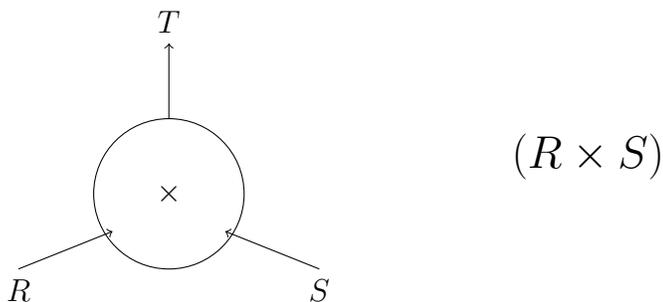
Le produit cartésien permet la construction de toutes les combinaisons possibles entre les n-uplets de 2 relations.

💣 Attention

la relation résultat du produit cartésien n'a pas de sens : on y associe en effet des attributs de relations différentes sans conserver de lien sémantique entre eux.

L'opération de jointure va, quant à elle, rétablir une relation résultat sémantiquement correcte.

3.4.1 Notation



Le produit cartésien est commutatif : $(R \times S)$ est équivalent à $(S \times R)$

3.4.2 Exemples

Exemple : 1

3.4 Produit cartésien

R	A	B	C
	a	b	c
	d	a	f
	c	b	d

S	A	D
	b	g
	d	a

$$T \leftarrow (R \times S)$$

T	R.A	R.B	R.C	S.A	S.D
	a	b	c	b	g
	a	b	c	d	a
	d	a	f	b	g
	d	a	f	d	a
	c	b	d	b	g
	c	b	d	d	a

Degré de la relation T : degré de R + degré de S.

Cardinalité de la relation T : cardinalité de R * cardinalité de S.

Exemple : 2

Etudiant	numero	nom	prénom	dateNaissance
	1	Dupont	Jacques	15/01/1990
	9	Durand	Pierre	20/07/1999
	23	Lambert	Paul	03/12/1994
	34	Durand	Jacques	15/12/1994
	105	Lambert	Paul	20/10/1995

Formation	code	intitule
	INFO	STS Informatique
	MATH	STS Mathematiques
	PC	STS Physique Chimie

$$T \leftarrow (Etudiant \times Formation)$$

3.4 Produit cartésien

T	numero	nom	prénom	dateNaissance	code	intitule
	1	Dupont	Jacques	15/01/1990	INFO	STS Informatique
	9	Durand	Pierre	20/07/1999	INFO	STS Informatique
	23	Lambert	Paul	03/12/1994	INFO	STS Informatique
	34	Durand	Jacques	15/12/1994	INFO	STS Informatique
	105	Lambert	Paul	20/10/1995	INFO	STS Informatique
	1	Dupont	Jacques	15/01/1990	MATH	STS Mathématiques
	9	Durand	Pierre	20/07/1999	MATH	STS Mathématiques
	23	Lambert	Paul	03/12/1994	MATH	STS Mathématiques
	34	Durand	Jacques	15/12/1994	MATH	STS Mathématiques
	105	Lambert	Paul	20/10/1995	MATH	STS Mathématiques
	1	Dupont	Jacques	15/01/1990	PC	STS Physique Chimie
	9	Durand	Pierre	20/07/1999	PC	STS Physique Chimie
	23	Lambert	Paul	03/12/1994	PC	STS Physique Chimie
	34	Durand	Jacques	15/12/1994	PC	STS Physique Chimie
	105	Lambert	Paul	20/10/1995	PC	STS Physique Chimie

Ce résultat n'a pas de sens : il fournit toutes les possibilités de combinaison de chaque étudiant avec toutes les formations...

4 Opérations dérivées

Elles peuvent être construites à partir des opérations de base.

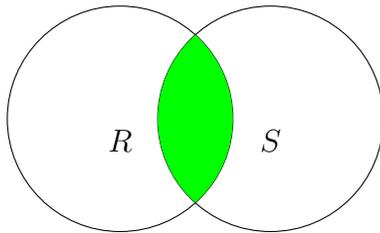
4.1 Intersection

○ Intersection

L'intersection de 2 relations R et S de même schéma est une relation T de même schéma (*union compatible*) contenant les n-uplets appartenant à la fois à R et à S, soit , soit : $(R \cap S) = \{t : t \in R \wedge t \in S\}$, opérateur logique ET.

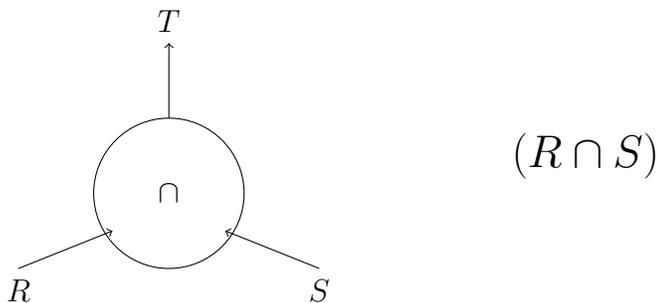
L'intersection peut être construite à partir de la différence :

$$(R \cap S) = R - (R - S) = S - (S - R)$$



(cf. Wikipedia : diagrammes d'Euler, de Venn et de Carroll)

4.1.1 Notation



L'intersection est commutative : $(R \cap S)$ est équivalent à $(S \cap R)$

4.1 Intersection

4.1.2 Exemples

Exemple : 1

R	A	B	C		S	A	B	C
	a	b	c	-		b	g	a
	d	a	f			d	a	f
	c	b	d					

$$T \leftarrow (R \cap S)$$

T	A	B	C
	d	a	f

Exemple : 2

Etudiant	numero	nom	prénom	dateNaissance
	1	Dupont	Jacques	15/01/1990
	9	Durand	Pierre	20/07/1999
	23	Lambert	Paul	03/12/1994
	34	Durand	Jacques	15/12/1994
	105	Lambert	Paul	20/10/1995

$$\sigma_{numero < 100} Etudiant$$

R1	numero	nom	prénom	dateNaissance
	1	Dupont	Jacques	15/01/1990
	9	Durand	Pierre	20/07/1999
	23	Lambert	Paul	03/12/1994
	34	Durand	Jacques	15/12/1994

$$\sigma_{prenom = "paul"} Etudiant$$

R2	numero	nom	prénom	dateNaissance
	23	Lambert	Paul	03/12/1994
	105	Lambert	Paul	20/10/1995

$$(\sigma_{numero < 100} Etudiant) \cap (\sigma_{prenom = "paul"} Etudiant)$$

T	numero	nom	prénom	dateNaissance
	23	Lambert	Paul	03/12/1994

4.2 Jointure interne

🔍 Jointure interne

| La jointure interne combine un produit cartésien et une sélection.

Elle va associer chacun des n-uplets d'une relation R avec tous les n-uplets d'une relation S qui respectent une qualification (= une condition).

Si un n-uplets de R ne trouve pas de correspondance dans S, il est éliminé du résultat.

La jointure interne est commutative : $(R \bowtie S)$ est équivalent à $(S \bowtie R)$

4.2.1 Thêta-jointure

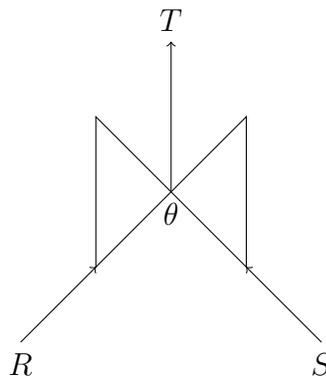
Cette opération est essentielle dans les systèmes relationnels pour rassembler de manière cohérente des attributs provenant de plusieurs relations.

La thêta-jointure est le résultat d'un produit cartésien de R et S qui conserve seulement les n-uplets satisfaisant une qualification (une qualification est une comparaison entre 2 attributs).

La jointure exige que les 2 relations aient au moins un domaine en commun.

Une équi-jointure est un cas particulier de thêta-jointure n'utilisant que l'opérateur d'égalité.

4.2.1.1 notation



$$(R \bowtie_Q S) \text{ ou : JOIN R (Q) S}$$

équivalent à :

$$(R \bowtie_Q S) = \sigma_Q(R \times S)$$

4.2 Jointure interne

Exemple : 1

R	A	B	C
	a	b	c
	d	a	f
	c	b	d

-

S	A	D
	b	g
	d	a

$$R1 \leftarrow (R \times S)$$

T	R.A	R.B	R.C	S.A	S.D
	a	b	c	b	g
	a	b	c	d	a
	d	a	f	b	g
	d	a	f	d	a
	c	b	d	b	g
	c	b	d	d	a

$$T \leftarrow \sigma_{R.B=S.A} R1$$

T	R.A	R.B	R.C	S.A	S.D
	a	b	c	b	g
	c	b	d	b	g

$$\text{soit : } T \leftarrow (R \bowtie_{R.B = S.A} S)$$

Exemple : 2

Etudiant	numero	nom	prénom	dateNaissance
	1	Dupont	Jacques	15/01/1990
	9	Durand	Pierre	20/07/1999
	23	Lambert	Paul	03/12/1994
	34	Durand	Jacques	15/12/1994
	105	Lambert	Paul	20/10/1995

4.2 Jointure interne

Inscrire	code	numero
	INFO	1
	INFO	9
	INFO	34
	MATH	1
	MATH	23
	MATH	34
	PC	1

soit : $T \leftarrow (Etudiant \bowtie Inscire)$
 $Etudiant.numero = inscire.numero$

T	numero	nom	prénom	dateNaissance	code	numero
	1	Dupont	Jacques	15/01/1990	INFO	1
	1	Dupont	Jacques	15/01/1990	MATH	1
	1	Dupont	Jacques	15/01/1990	PC	1
	9	Durand	Pierre	20/07/1999	INFO	9
	23	Lambert	Paul	03/12/1994	MATH	23
	34	Durand	Jacques	15/12/1994	INFO	34
	34	Durand	Jacques	15/12/1994	MATH	34

4.2.2 Jointure naturelle

La jointure naturelle est une jointure interne d'égalité de R et S dont le prédicat porte sur l'égalité de tous les attributs de même nom, suivi d'une projection qui permet de supprimer les attributs répétés.



Attention à l'utilisation de cette forme de jointure

- si aucun nom d'attribut n'est commun aux 2 relations, on obtient un produit cartésien
- si des attributs portent le même nom sans toutefois avoir le même sens, on obtient une jointure incohérente...(cela ne devrait pas se produire, car des attributs ayant un sens différent porteront sur des domaines différents et des valeurs de domaines différents ne

4.2 Jointure interne

peuvent être comparées)

Exemple : 1

R	A	B	C
a	b	c	
d	a	f	
c	b	d	

-

S	A	D
b	g	
d	a	

$$R1 \leftarrow (R \times S)$$

T	R.A	R.B	R.C	S.A	S.D
	a	b	c	b	g
	a	b	c	d	a
	d	a	f	b	g
	d	a	f	d	a
	c	b	d	b	g
	c	b	d	d	a

$$R2 \leftarrow \sigma_{R.A=S.A}(R1)$$

T	R.A	R.B	R.C		S.D
	d	a	f	d	a

$$T \leftarrow \pi_{R.A,R.B,R.C,S.D}(R2)$$

T	R.A	R.B	R.C	S.D
	d	a	f	a

équivalent à : $T \leftarrow (R \bowtie S)$

Exemple : 2

Etudiant	numero	nom	prénom	dateNaissance
	1	Dupont	Jacques	15/01/1990
	9	Durand	Pierre	20/07/1999
	23	Lambert	Paul	03/12/1994
	34	Durand	Jacques	15/12/1994
	105	Lambert	Paul	20/10/1995

4.3 Jointure externe

Inscrire	code	numero
	INFO	1
	INFO	9
	INFO	34
	MATH	1
	MATH	23
	MATH	34
	PC	1

Soit $T \leftarrow (Etudiant \bowtie inscrire)$

T	numero	nom	prénom	dateNaissance	code
	1	Dupont	Jacques	15/01/1990	INFO
	1	Dupont	Jacques	15/01/1990	MATH
	1	Dupont	Jacques	15/01/1990	PC
	9	Durand	Pierre	20/07/1999	INFO
	23	Lambert	Paul	03/12/1994	MATH
	34	Durand	Jacques	15/12/1994	INFO
	34	Durand	Jacques	15/12/1994	MATH



Jointure naturelle contrainte

On privilégiera la forme de jointure naturelle en précisant la colonne de jointure afin de simplifier les opérations de renommage qui alourdissent inutilement les expressions relationnelles complexes.

Ainsi pour deux relations $R(A,B,C)$ et $S(D,A)$, on utilisera la forme suivante :

$R \underset{A}{\bowtie} S$, soit la jointure des relations R et S sur l'égalité de leur attribut commun 'A'.

4.3 Jointure externe

La jointure externe (OUTER JOIN) est une jointure de R et S qui conserve

- les n-uplets résultat de la jointure

4.3 Jointure externe

- ainsi que les n-uplets de l'une ou l'autre des 2 relations, ou bien encore des 2, qui n'ont pas été joints.

La jointure de R et S conserve tous les n-uplets :

- de R pour une jointure externe gauche, \bowtie (LEFT OUTER JOIN)
- de S pour une jointure externe droite, \bowtie (RIGHT OUTER JOIN)
- ou des 2 relations pour une jointure totale, \bowtie (FULL OUTER JOIN).

La valeur des attributs des n-uplets de la relation ne satisfaisant pas au prédicat/qualification ont pour valeur "NULL".

Exemple : 1

R	A	B	C
a	b	c	
d	a	f	
c	b	d	

S	A	D
b	g	
d	a	

$$R1 \leftarrow (R \times S)$$

T	R.A	R.B	R.C	S.A	S.D
	a	b	c	b	g
	a	b	c	d	a
	d	a	f	b	g
	d	a	f	d	a
	c	b	d	b	g
	c	b	d	d	a

T	R.A	R.B	R.C	S.A	S.D
	a	b	c	null	null
	d	a	f	d	a
	c	b	d	null	null

$$soit : T \leftarrow (R \bowtie_{R.A = S.A} S)$$

Exemple : 2

4.4 Cas particuliers de jointures

Etudiant	numero	nom	prénom	dateNaissance
	1	Dupont	Jacques	15/01/1990
	9	Durand	Pierre	20/07/1999
	23	Lambert	Paul	03/12/1994
	34	Durand	Jacques	15/12/1994
	105	Lambert	Paul	20/10/1995

Inscrire	code	numero
	INFO	1
	INFO	9
	INFO	34
	MATH	1
	MATH	23
	MATH	34
	PC	1

$$T \leftarrow (Etudiant \bowtie_{Etudiant.numero = inscrire.numero} Inscire)$$

T	numero	nom	prénom	dateNaissance	code	numero
	1	Dupont	Jacques	15/01/1990	INFO	1
	1	Dupont	Jacques	15/01/1990	MATH	1
	1	Dupont	Jacques	15/01/1990	PC	1
	9	Durand	Pierre	20/07/1999	INFO	9
	23	Lambert	Paul	03/12/1994	MATH	23
	34	Durand	Jacques	15/12/1994	INFO	34
	34	Durand	Jacques	15/12/1994	MATH	34
	105	Lambert	Paul	20/10/1995	null	null

4.4 Cas particuliers de jointures

4.4.1 Auto-jointure

L'auto-jointure est un cas de jointure d'une relation avec elle-même, quelle soit interne ou externe.

Cela se passe comme si on avait 2 copies différentes d'une même relation.

La relation doit être renommée pour être utilisée 2 fois dans l'opération de jointure. Les noms des attributs doivent être préfixés du nom de relation

afin d'éviter toute ambiguïté ou bien être renommés.

4.4.1.1 Exemple à partir de la relation *Etudiant*, produire une relation comportant tous les binômes possibles d'une même ville :

$$T \leftarrow (Etudiant \underset{\text{ville}}{\bowtie} \rho_V(Etudiant))$$

Une amélioration sera à apporter : en effet, chaque étudiant formera un binôme avec lui-même...

4.4.2 Semi-jointure

La semi-jointure est une jointure interne à laquelle est appliquée une projection pour ne conserver que les attributs d'une des 2 relations.

Semi-jointure gauche : joint R à S et ne conserve que les attributs de R

$$(R \ltimes S)$$

Semi-jointure droite : joint R à S et ne conserve que les attributs de S

$$(R \rtimes S)$$

4.4.3 Anti-jointure

L'anti-jointure est une opération de jointure qui ne conserve que les tuples qui n'ont pas été joints d'une des 2 relations.

Anti-jointure gauche : joint R à S, et ne conserve que les attributs de R et les tuples de R qui n'ont pas été joints.

$$(R \triangleright S)$$

Anti-jointure droite : joint R à S, et ne conserve que les attributs de S et les tuples de S qui n'ont pas été joints.

$$(R \triangleleft S)$$

4.5 Division

Le quotient (ou division) : de la relation R de schéma $R(A_1, A_2, \dots, A_n)$ par la sous-relation S de schéma $S(A_{p+1}, \dots, A_n)$ est la relation T de schéma $T(A_1, A_2, \dots, A_p)$ formée de tous les n -uplets qui, combinés à chaque n -uplets de S , donnent toujours un n -uplets de R , soit $(R \div S) = \{t : \forall s \in S, (t, s) \in R\}$.

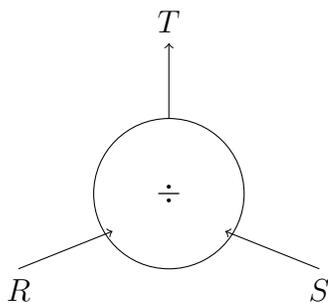
La construction du quotient passe par les étapes suivantes :

- R1 : projection sur les attributs de R qui n'appartiennent pas à S
- R2 : produit cartésien de la relation R1 avec S
- R3 : différence de cette R2 - R
- R4 : projection des attributs de R3 sur les attributs qui appartiennent à R et pas à S
- T : différence R1 - R4

soit :

$$R \div S = \pi_x(R) - \pi_x((\pi_x(R) \times S) - R)$$

4.5.1 Notation



$(R \div S)$
 équivalent à :
 $(R \div S) = \pi_x(R) - \pi_x(\pi_x(R) \times S) - R$

4.5.2 Exemples

Exemple : 1

R	A	B	C	D
	a	b	c	d
	a	b	a	d
	a	c	b	a
	b	c	c	d
	d	a	c	d
	b	c	a	d

S	C	D
	c	d
	a	d

$$T \leftarrow (R \div S)$$

T	A	B
	a	b
	b	c

Le produit cartésien de T et S donnera un n-uplet qui appartient à R.

Vérification :

- R1 : projection sur les attributs de R qui n'appartiennent pas à S

R1	A	B
	a	b
	a	c
	b	c
	d	a

- R2 : produit cartésien de la relation R1 avec S

R2	A	B	C	D
	a	b	c	d
	a	c	c	d
	b	c	c	d
	d	a	c	d
	a	b	a	d
	a	c	a	d
	b	c	a	d
	d	a	a	d

- R3 : $(R2 - R)$

R3	A	B	C	D
	a	c	c	d
	a	c	a	d
	d	a	a	d

- R4 : projection des attributs de R3 sur les attributs qui appartiennent à R et pas à S

R4	A	B
	a	c
	d	a

- T : $(R1 - R4)$

T	A	B
	a	b
	b	c

Le quotient répond à des questions comme :

- les produits qui existent dans une gamme de couleurs(ou une gamme de prix),
- les livres qu'on trouvera dans une gamme de reliures,
- les coureurs qui ont participé à une liste de grands prix, etc.
- les étudiants qui ont été présents à toutes les séances, etc.
- les étudiants qui sont inscrits à toutes les modules de formation, etc.

Exemple : 2

Les numéro d'étudiants qui sont inscrits à tous les codes de formations :

Inscrire	code	numero
	INFO	1
	INFO	9
	INFO	34
	MATH	1
	MATH	23
	MATH	34
	PC	1

$$R1 \leftarrow \pi_{code}(Formation)$$

R1	code
	INFO
	MATH
	PC

$$T \leftarrow (Inscrire \div R1)$$

T	numero
	1

Le produit cartésien de R1 et T donnera un n-uplet qui appartient à Inscire.

5 Extensions

Des extensions ont été proposées dans les années 1980 afin de prendre en compte des cas et questions plus complexes.

5.1 Agrégats de n-uplets

L'agrégat appliqué à des attributs correspond à l'application de fonctions statistiques

- de manière globale pour tous les n-uplets d'une relation.
- ou en définissant une liste d'attributs de regroupement des n-uplets

Les opérations statistiques applicables sont les suivantes :

- count : dénombrer les n-uplets d'une relation
- sum : cumuler les valeurs d'un attribut
- avg (average) : calculer la moyenne des valeurs d'un attribut
- min : déterminer la plus petite des valeurs d'un attribut
- max : déterminer la plus grande des valeurs d'un attribut

5.1.1 Agrégat global

5.1.1.1 Notation $\mathcal{G}_{fonction(x1)}(R)$

5.1.1.2 Exemples

Exemple : 1

R	X	Y	N
	a	b	10
	a	c	30
	b	b	20
	c	b	20

$$T \leftarrow \mathcal{G}_{count(X)}(R)$$

T	count(X)
	4

$$T \leftarrow \mathcal{G}_{sum(N)}(R)$$

T	sum(N)
	80

5.1.2 Agrégat par valeur de regroupement

$$a_1, a_2, \dots, a_n \mathcal{G}_{fonction(x_1)}(R)$$

5.1.2.1 Exemples

Exemple : 1

R	X	Y	N
	a	b	10
	a	c	30
	b	b	20
	c	b	20

$$T \leftarrow_X \mathcal{G}_{count(X)}(R)$$

T	X	count(X)
	a	2
	b	1
	c	1

$$T \leftarrow_X \mathcal{G}_{Sum(N)}(R)$$

T	X	sum(N)
	a	40
	b	20
	c	20

5.2 Calculs de valeurs d'attributs

Cette opération ne fait pas partie des opérations de l'algèbre relationnelle initiale.

L'opération de projection étendue permet le calcul de nouvelles valeurs d'attributs à partir des attributs et d'opérateurs arithmétiques. Ces valeurs calculées peuvent être renommées.

Exemple : 1

R	A	B	C
	a	10	1.5
	d	12	2
	c	50	0.5

$$\pi_{A,B,B*C \rightarrow \text{montant}}(R)$$

R	A	B	montant
	a	10	15
	d	12	24
	c	50	25

5.3 Classement des n-uplets

Cette opération ne fait pas partie des opérations de l'algèbre relationnelle initiale.

Une opération de classement permet d'ordonner les n-uplets selon certains critères.

$$T \leftarrow O_{x_1 \text{critre}, \dots, x_n \text{critre}}(R)$$

où critère est

- soit ASC (ASCending, ordre croissant)
- soit DESC (DESCending, ordre décroissant)

qui produit une nouvelle relation T à partir de la relation R, en classant les n-uplets selon les critères de tri.

5.4 Opérations de mise à jour

Les opérations de mise à jour des n-uplets peuvent être traitées grâce aux opérateurs de l'algèbre relationnelle.

5.4.1 Suppression de n-uplets

$$R \leftarrow R - \sigma_Q(R)$$

5.4.2 Insertion de n-uplets

$$R \leftarrow R \cup (x_1, x_2, \dots, x_n)$$

6 Expressions relationnelles

6.1 Composition des opérations

Une fois les opérateurs relationnels identifiés, il existe une combinaison des opérations des calculs qui permet de fournir un résultat optimal. Le coût d'une expression relationnelle est généralement associé à la quantité de données manipulées (degré et cardinalité des relations).

Il existe en général plusieurs requêtes algébriques différentes qui fournissent une solution à une question donnée. Si le résultat est le même, l'efficacité l'est rarement (par exemple, l'opération de jointure est en général coûteuse ; la relation temporaire la plus volumineuse pour effectuer cette requête est engendrée par le produit cartésien des 2 tables).

Des calculs utilisant la cardinalité d'une relation ainsi que le nombre et la taille (moyenne) de ses attributs permet de choisir les requêtes la moins coûteuse et d'ainsi optimiser la performance des requêtes.

L'assignation du résultat peut être réalisé dans une relation temporaire.

6.2 Équivalences

Savoir...

Les sélections successives sont commutatives :

$$\sigma_{e1}(\sigma_{e2}(R)) \text{ est équivalent à : } \sigma_{e2}(\sigma_{e1}(R))$$

$e1$ et $e2$ sont des expressions logiques.

Attention!!!

Les projections successives ne sont généralement pas commutatives.

Les projections successives NE sont commutatives QUE SI dans l'expression suivante :

$$\pi_{l2}(\pi_{l1}(R1)) \text{ , seulement si } l2 \subset l1$$

l1 et l2 sont des listes d'attributs.

 **Savoir...**

Les jointures successives sont associatives :

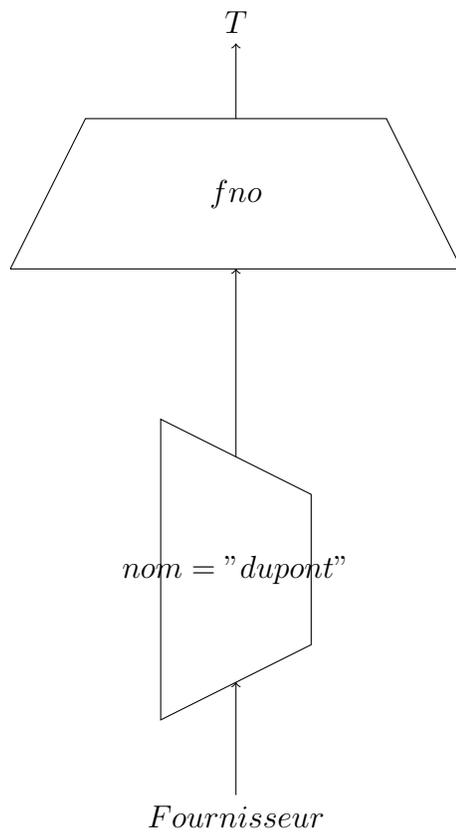
$(R \bowtie (S \bowtie T))$ est équivalent à $((R \bowtie S) \bowtie T)$

6.3 Représentation arborescente

L'arbre algébrique est une représentation graphique obtenu en utilisant l'enchaînement des opérations de calcul d'une expression :

- les feuilles représentent les relations
- les noeuds correspondent aux opérateurs

Exemple : Déterminer les différents numéros de fournisseurs nommés "Dupont" (cf. [9](#) page [52](#)) :



$$R1 \leftarrow \sigma_{nom="dupont"}(Fournisseur)$$

$$T \leftarrow \pi_{fno}(R1)$$

7 Estimation du cout d'une expression relationnelle

L'estimation du cout va être basé, de manière simpliste, sur le produit du nombre de tuples (cardinalité) par le degré (nombre d'attributs) des relations mises en oeuvre. Pour obtenir une précision supplémentaire, il faudra utiliser l'espace moyen occupé par attribut selon son domaine.

Généralement, on placera la sélection au plus tôt, cet opérateur étant très réducteur en terme de nombre de tuples. Les jointures seront placées ensuite. Les projections seront enfin placées au plus tard, lorsque tous les attributs utiles aux sélection et jointures auront été utilisés.

Cependant, cela ne garantit pas toujours d'obtenir une requête optimale...

7.1 Selection

La sélection parcourt tous les tuples de la relation et effectue un test sur chacun d'eux

- complexité : $\text{card}(R)$
- taille du résultat : $\text{degré}(R) \times [0, \text{card}(R)]$

7.2 Projection

La projection parcourt tous les tuples et élimine les doublons

- complexité : $\text{card}(R) + \text{complexité de recherche des doublons}$
- taille du résultat : $\text{degré}(\text{resultat}) \times \text{card}(R)$

7.3 Produit cartésien

Le produit cartésien parcourt tous les n-uplets de la 1ère relation et, pour chacun, tous les n-uplets de la 2nde relation

- complexité : $\text{card}(R) \times \text{card}(S)$
- taille du résultat : $(\text{degré}(R) + \text{degré}(S)) \times (\text{card}(R) \times \text{card}(S))$

7.4 Thétajointure

La thétajointure effectue un produit cartésien, puis elle sélectionne les tuples correspondant aux critères de jointure

- complexité : $\text{card}(R) \times \text{card}(S)$
- taille du résultat (si clef étrangère de S vers R) : $(\text{degré}(R) + \text{degré}(S)) \times \text{card}(S)$

7.5 Jointure naturelle

La jointure effectue un produit cartésien, puis elle sélectionne les tuples correspondant aux critères de jointure

- complexité : $\text{card}(R) \times \text{card}(S)$
- taille du résultat (si clef étrangère de S vers R) : $(\text{degré}(R) + \text{degré}(S) - \text{nombre de colonnes de jointure}) \times \text{card}(S)$

7.6 Exemple

Soient les 3 relations suivantes :

- Fournisseur(idFou, nomFou, villeFou)
- Commande (#idFou,#idPro, qte)
- Produit (idPro, nomPro, prixPro)

et une volumétrie estimée en nombre de tuples :

- Fournisseur : 100
- Commande 10000
- Produit : 500

L'obtention des noms des fournisseurs du produit "p1" va nécessiter les opérations suivantes :

7.6.0.1 Version 1

1. $R1 \leftarrow (Fournisseur \bowtie Commande)$: 10000 x 100 tuples lus pour 10000 tuples résultat
2. $R2 \leftarrow \sigma_{\#idPro="p1"}(R1)$: 10000 tuples lus pour 200 tuples résultat (estimé)
3. Final $\leftarrow \pi_{nomFou}(R2)$: 200 tuples lus pour 30 tuples résultat (estimé)

soit au total, en nombre d'opérations de lecture de tuples : $10000 \times 100 + 10000 + 200 = 1010200$

7.6.0.2 Version 2, optimisée

1. $R1 \leftarrow \sigma_{\#idPro=p1}(Commande)$: 10000 tuples lus pour 200 tuples résultat (estimé)
2. $R2 \leftarrow (Fournisseur \bowtie R1)$: 200 x 100 tuples lus pour 200 tuples résultat
3. $Final \leftarrow \pi_{nomFou}(R2)$: 200 tuples lus pour 30 tuples résultat (estimé)

soit au total, en nombre d'opérations de lecture de tuples : $10000 + 20000 + 200 = 30200$

8 Mémo - Opérateurs de l'algèbre relationnelle

symbole	operation	exemple
π	projection	$\pi_{x_1, \dots, x_n}(R)$
σ	sélection	$\sigma_Q(R)$
ρ	renommage	$\rho_S(R)$
ρ	renommage	$\rho_{(x_1/nx_1, x_2/nx_2, \dots)}(R)$
\times	produit cartésien	$(R \times S)$
$-$	différence	$(R - S)$
\cup	union	$(R \cup S)$
\cap	intersection	$(R \cap S)$
\bowtie	jointure naturelle	$(R \bowtie S)$
\bowtie_Q	théta jointure	$(R \bowtie_Q S)$
\bowtie_Q	jointure ext. gauche	$(R \bowtie_Q S)$
\bowtie_Q	jointure ext. droite	$(R \bowtie_Q S)$
\bowtie_Q	jointure ext. complete	$(R \bowtie_Q S)$
\ltimes	semi-jointure gauche	$(R \ltimes S)$
\rtimes	semi-jointure droite	$(R \rtimes S)$
\triangleright	anti-jointure gauche	$(R \triangleright S)$
\triangleleft	anti-jointure droite	$(R \triangleleft S)$
\div	quotient	$(R \div S)$
\mathcal{G}	agrégat global	$\mathcal{G}_{F(A)}(R)$
\mathcal{G}	agrégat regroupement	$x_1, x_2, \dots \mathcal{G}_{F(A)}(R)$

9 Exercices

Soit le schéma de relation suivant :

- Fournisseur (fno, nom, adresse, ville)
- Produit (pno, design, prix, poids, couleur)
- Commande (cno, #pno, #fno, qte)

9.1 Questions

1. lister les numéros et noms des fournisseurs
2. lister les données sur les produits dont le poids est supérieur à 15kg
3. liste les produits dont le poids est compris entre 15 et 40 kg
4. lister les fournisseurs de Lille, Lyon ou Nice
5. lister les noms des fournisseurs avec les numéros de produits commandés ainsi que la quantité commandée
6. lister les couples de références de fournisseurs situés dans la même ville
7. lister tous les produits de moins de 20 kg avec les quantités en cours de commande
8. compter le nombre de commandes du produit no 102
9. lister le nombre de commandes par fournisseur
10. lister les numéros de fournisseur qui ont plus de 3 commandes d'au moins dix articles en cours

9.2 Propositions de réponses :

1. lister les numéros et noms des fournisseurs
 - (a) $\pi_{fno,nom}(Fournisseur)$
2. lister les données sur les produits dont le poids est supérieur à 15kg
 - (a) $\sigma_{poids>15}(Produit)$
3. liste les produits dont le poids est compris entre 15 et 40 kg
 - (a) $R1 \leftarrow \sigma_{poids \geq 15}(Produit)$
 - (b) $R2 \leftarrow \sigma_{poids \leq 40}(Produit)$
 - (c) $R3 \leftarrow (R1 \cap R2)$
4. lister les fournisseurs de Lille, Lyon ou Nice
 - (a) $R1 \leftarrow \sigma_{ville="Lille"}(Fournisseurs)$
 - (b) $R2 \leftarrow \sigma_{ville="Lyon"}(Fournisseurs)$
 - (c) $R3 \leftarrow \sigma_{ville="Nice"}(Fournisseurs)$
 - (d) $R4 \leftarrow (R1 \cup R2)$
 - (e) $R5 \leftarrow (R4 \cup R3)$
5. lister les noms des fournisseurs avec les numéros de produits commandés ainsi que la quantité commandée
 - (a) $R1 \leftarrow_{fno,pno} \mathcal{G}_{sum(qte)/sommeQte}(Commande)$
 - (b) $R2 \leftarrow (Fournisseurs \bowtie R1)$
 - (c) $R3 \leftarrow \pi_{nom,pno,sommeQte}(R2)$
6. lister les couples de références de fournisseurs situés dans la même ville
 - (a) $(Fournisseurs \bowtie_{ville, fno} Fournisseur)$
7. lister tous les produits de moins de 20 kg avec les quantités en cours de commande
 - (a) $R1 \leftarrow \sigma_{poids < 20}(Produit)$

RÉFÉRENCES

- (b) $R2 \leftarrow_{pno} \mathcal{G}_{sum(qte)/sommeQte}(Commande)$
 - (c) $R3 \leftarrow (R1 \bowtie_{pno} R2)$
8. compter le nombre de commandes du produit no 102
- (a) $R1 \leftarrow \sigma_{pno=102}(Commande)$
 - (b) $R2 \leftarrow_{count(pno):nbProduits} (R1)$
9. lister le nombre de commandes par fournisseur
- (a) $R1 \leftarrow_{fno} \mathcal{G}_{count(cno)/nbCommandes}(Commande)$
 - (b) $R2 \leftarrow (Fournisseur \bowtie_{fno} R1)$
 - (c) $R3 \leftarrow \pi_{fno,nom,nbCommandes}(R2)$
10. lister les numéros de fournisseur qui ont plus de 3 commandes
- (a) $R1 \leftarrow_{fno} \mathcal{G}_{count(cno)/nbCommandes}(Commande)$
 - (b) $R2 \leftarrow \sigma_{nbCommandes>3}(R1)$
 - (c) $R3 \leftarrow \pi_{fno}(R2)$

Références

- [Codd, 1969] Codd, E. F. (1969). Derivability, redundancy and consistency of relations stored in large data banks. *IBM Research Report, San Jose, California*.
- [Codd, 1970] Codd, E. F. (1970). A relational model of data for large shared data banks. *Commun. ACM*, 13(6) :377–387.
- [Codd, 1971] Codd, E. F. (1971). Normalized data base structure : a brief tutorial. In *Proceedings of the 1971 ACM SIGFIDET (now SIGMOD) Workshop on Data Description, Access and Control*, SIGFIDET '71, pages 1–17, New York, NY, USA. ACM.

Index

avg, [41](#)

count, [41](#)

max, [41](#)

min, [41](#)