Chapitre 5

L'algèbre relationnelle

Sommaire

5.1 Les opérateurs de l'algèbre relationnelle 54
5.1.1 La sélection, σ 55
5.1.2 La projection, π 55
5.1.3 Le produit cartésien, $\times$ 56
5.1.4 L'union, $\cup$ 57
5.1.5 La différence, $-$ 58
5.1.6 Jointure, $\bowtie$ 58
5.2 Expression de requêtes avec l'algèbre 59
5.2.1 Sélection généralisée 59
5.2.2 Requêtes conjonctives 60
5.2.3 Requêtes avec $\cup$ et $-$ 61
5.3 Exercices 62

Le premier langage étudié dans ce cours est *l'algèbre relationnelle*. Il consiste en un ensemble d'opérations qui permettent de manipuler des *relations*, considérées comme des ensemble de tuples : on peut ainsi faire *l'union* ou la *différence* de deux relations, *sélectionner* une partie de la relation, effectuer des *produits cartésiens* ou des *projections*, etc.

Une propriété fondamentale de chaque opération est qu'elle prend une ou deux relations en entrée, et produit une relation en sortie. Cette propriété permet de *composer* des opérations : on peut appliquer une sélection au résultat d'un produit cartésien, puis une projection au résultat de la sélection et ainsi de suite. En fait on peut construire des *expressions algébriques* arbitrairement complexes qui permettent d'exprimer des manipulations sur un grand nombre de relations.

Une *requête* est une expression algébrique qui s'applique à un ensemble de relations (la base de données) et produit une relation finale (le résultat de la requête). On peut voir l'algèbre relationnelle comme un langage de programmation très simple qui permet d'exprimer des requêtes sur une base de données relationnelle.

Dans tout ce chapitre on va prendre l'exemple de la (petite) base de données d'un organisme de voyage. Cet organisme propose des séjours (sportifs, culturels, etc) se déroulant dans des stations de vacances. Chaque station propose un ensemble d'activtés (ski, voile, tourisme). Enfin on maintient une liste des clients (avec le solde de leur compte !) et des séjours auxquels ils ont participé avec leurs dates de début et de fin.

- Station (**nomStation**, capacité, lieu, région, tarif)
- Activite (***nomStation***, **libellé**, prix)
- Client (**id**, nom, prénom, ville, région, solde)
- Séjour (***idClient***, **station**, **début**, nbPlaces)

nomStation	capacité	lieu	région	tarif
Venusa	350	Guadeloupe	Antilles	1200
Farniente	200	Seychelles	Océan Indien	1500
Santalba	150	Martinique	Antilles	2000
Passac	400	Alpes	Europe	1000

La table *Station*

NomStation	Libellé	Prix
Venusa	Voile	150
Venusa	Plongée	120
Farniente	Plongée	130
Passac	Ski	200
Passac	Piscine	20
Santalba	Kayac	50

La table *Activité*

FIG. 5.1 – *Les stations et leurs activités*

id	nom	prénom	ville	région	solde
10	Fogg	Phileas	Londres	Europe	12465
20	Pascal	Blaise	Paris	Europe	6763
30	Kerouac	Jack	New York	Amérique	9812

La table *Client*

idClient	station	début	nbPlaces
10	Passac	1998-07-01	2
30	Santalba	1996-08-14	5
20	Santalba	1998-08-03	4
30	Passac	1998-08-15	3
30	Venusa	1998-08-03	3
20	Venusa	1998-08-03	6
30	Farniente	1999-06-24	5
10	Farniente	1998-09-05	3

La table *Séjour*

FIG. 5.2 – *Les clients et leurs séjours*

5.1 Les opérateurs de l'algèbre relationnelle

L'algèbre se compose d'un ensemble d'opérateurs, parmi lesquels 5 sont nécessaires et suffisants et permettent de définir les autres par composition. Ce sont :

1. La sélection, dénotée σ ;
2. La projection, dénotée π ;
3. Le produit cartésien, dénoté $\times$;
4. L'union, $\cup$;
5. La différence, $-$.

Les deux premiers sont des opérateurs *unaires* (ils prennent en entrée une seule relation) et les autres sont des opérateurs *binaires*. A partir de ces opérateurs il est possible d'en définir d'autres, et notamment la *jointure*, $\bowtie$, qui est la composition d'un produit cartésien et d'une sélection.

Ces opérateurs sont maintenant présentés tour à tour.

5.1.1 La sélection, σ

La sélection $\sigma_F(R)$ s'applique à une relation, R, et extrait de cette relation les tuples qui satisfont un critère de sélection, F. Ce critère peut être :

- La comparaison entre un attribut de la relation, A, et une constante a. Cette comparaison s'écrit $A\Theta a$, où Θ appartient à $\{=, <, >, \leqslant, \geqslant\}$.
- La comparaison entre deux attributs A_1 et A_2, qui s'écrit $A_1\Theta A_2$ avec les mêmes opérateurs de comparaison que précédemment.

Premier exemple : exprimer la requête qui donne toutes les stations aux Antilles.

$$\sigma_{region='Antilles'}(Station)$$

On obtient donc le résultat :

nomStation	capacité	lieu	région	tarif
Venusa	350	Guadeloupe	Antilles	1200
Santalba	150	Martinique	Antilles	2000

La sélection a pour effet de supprimer des lignes, mais chaque ligne garde l'ensemble de ses attributs.

5.1.2 La projection, π

La projection $\pi_{A_1,A_2,...,A_k}(R)$ s'applique à une relation R, et ne garde que les attributs $A_1, A_2, \ldots A_k$. Donc, contrairement à la sélection, on ne supprime pas des lignes mais des colonnes.

Exemple : donner le nom des stations, et leur région.

$$\pi_{nomStation,region}(Station)$$

On obtient le résultat suivant, après suppression des colonnes `capacité`, `lieu` et `tarif` :

nomStation	région
Venusa	Antilles
Farniente	Océan Indien
Santalba	Antilles
Passac	Europe

En principe le nombre de lignes dans le résultat est le même que dans la relation initiale. Il y a cependant une petite subtilité : comme le résultat est une relation, il ne peut pas y avoir deux lignes identiques (il n'y a pas deux fois le même élément dans un ensemble). Il peut arriver qu'après une projection, deux lignes qui étaient distinctes initialement se retrouvent identiques, justement parce ce que l'attribut qui permettait de les distinguer a été supprimé. Dans ce cas on ne conserve qu'une seule des deux (ou plus) lignes identiques.

Exemple : on souhaite connaître toutes les régions où il y a des stations. On exprime cette requête par :

$$\pi_{region}(Station)$$

et on obtient :

région
Antilles
Océan Indien
Europe

La ligne 'Antilles' était présente deux fois dans la relation *Station*, et n'apparaît plus qu'en un seul exemplaire dans le résultat.

5.1.3 Le produit cartésien, $\times$

Le premier opérateur binaire, et le plus important, est le produit cartésien, $\times$. Le produit cartésien entre deux relations R et S se note $R \times S$, et permet de créer une nouvelle relation où chaque tuple de R est associé à chaque tuple de S.

Voici deux relations R et S :

A	B
a	b
x	y

C	D
c	d
u	v
x	y

Et voici le résultat de $R \times S$:

A	B	C	D
a	b	c	d
a	b	u	v
a	b	x	y
x	y	c	d
x	y	u	v
x	y	x	y

Le nombre de lignes dans le résultat est exactement $|R| \times |S|$ ($|R|$ dénote le nombre de lignes dans la relation R).

En lui-même, le produit cartésien ne présente pas un grand intérêt puisqu'il associe aveuglément chaque ligne de R à chaque ligne de S. Il ne prend vraiment son sens qu'associé à l'opération de sélection, ce qui permet d'exprimer des *jointures*, opération fondamentale qui sera détaillée plus loin.

Conflits de noms d'attributs

Quand les schémas des relations R et S sont complêtement distincts, il n'y a pas d'ambiguité sur la provenance des colonnes dans le résultat. Par exemple on sait que les valeurs de la colonne A dans $R \times S$ viennent de la relation R. Il peut arriver (il arrive de fait très souvent) que les deux relations aient des attributs qui ont le même nom. On doit alors se donner les moyens de distinguer l'origine des colonnes dans la table résultat en donnant un nom distinct à chaque attribut.

Voici par exemple une table T qui a les mêmes noms d'attributs que R. Le schéma du résultat du produit cartésien $R \times T$ a pour schéma (A, B, A, B) et présente donc des ambiguités, avec les colonnes A et B en double.

A	B
m	n
o	p

La table T

La première solution pour lever l'ambiguité est de préfixer un attribut par le nom de la table d'où il provient. Le résultat de $R \times T$ devient alors :

R.A	R.B	T.A	T.B
a	b	m	n
a	b	n	p
x	y	m	n
x	y	n	p

Au lieu d'utiliser le nom de la relation en entier, on peut s'autoriser tout synonyme qui permet de lever l'ambiguité. Par exemple, dans le produit cartésien `Station` $\times$ `Activite`, on peut utiliser `S` comme préfixe pour les attributs venant de `Station`, et `A` pour ceux venant d'`Activite`. Le résultat peut alors être représenté comme sur la Fig. 5.3. On lève l'ambiguité sur les attributs `nomStation` qui apparaissent deux fois. Ce n'est pas nécessaire pour les autres attributs.

S.nomStation	cap.	lieu	région	tarif	A.nomStation	libelle	prix
Venusa	350	Guadeloupe	Antilles	1200	Venusa	Voile	150
Venusa	350	Guadeloupe	Antilles	1200	Venusa	Plongée	120
Venusa	350	Guadeloupe	Antilles	1200	Farniente	Plongée	130
Venusa	350	Guadeloupe	Antilles	1200	Passac	Ski	200
Venusa	350	Guadeloupe	Antilles	1200	Passac	Piscine	20
Venusa	350	Guadeloupe	Antilles	1200	Santalba	Kayac	50
Farniente	200	Seychelles	Océan Indien	1500	Venusa	Voile	150
Farniente	200	Seychelles	Océan Indien	1500	Venusa	Plongée	120
Farniente	200	Seychelles	Océan Indien	1500	Farniente	Plongée	130
Farniente	200	Seychelles	Océan Indien	1500	Passac	Ski	200
Farniente	200	Seychelles	Océan Indien	1500	Passac	Piscine	20
Farniente	200	Seychelles	Océan Indien	1500	Santalba	Kayac	50
Santalba	150	Martinique	Antilles	2000	Venusa	Voile	150
Santalba	150	Martinique	Antilles	2000	Venusa	Plongée	120
Santalba	150	Martinique	Antilles	2000	Farniente	Plongée	130
Santalba	150	Martinique	Antilles	2000	Passac	Ski	200
Santalba	150	Martinique	Antilles	2000	Passac	Piscine	20
Santalba	150	Martinique	Antilles	2000	Santalba	Kayac	50
Passac	400	Alpes	Europe	1000	Venusa	Voile	150
Passac	400	Alpes	Europe	1000	Venusa	Plongée	120
Passac	400	Alpes	Europe	1000	Farniente	Plongée	130
Passac	400	Alpes	Europe	1000	Passac	Ski	200
Passac	400	Alpes	Europe	1000	Passac	Piscine	20
Passac	400	Alpes	Europe	1000	Santalba	Kayac	50

FIG. 5.3 – *Produit cartésien* Station × Activité, *avec levée des ambiguïtés*

Renommage

Il existe une deuxième possibilité pour résoudre les conflits de noms : le *renommage*. Il s'agit d'un opérateur particulier, dénoté ρ, qui permet de renommer un ou plusieurs attributs d'une relation. L'expression $\rho_{A\to C,B\to D}(T)$ permet ainsi de renommer A en C et B en D dans la relation T. Le produit cartésien

$$R \times \rho_{A\to C,B\to D}(T)$$

ne présente alors plus d'ambiguités. Le renommage est une solution très générale, mais asez lourde à utiliser

Il est tout à fait possible de faire le produit cartésien d'une relation avec elle-même. Dans ce cas le renommage où l'utilisation d'un préfixe distinctif est impératif. Voici par exemple le résultat de $R \times R$, dans lequel on préfixe par $R1$ et $R2$ respectivement les attributs venant de chacune des opérandes.

R1.A	R1.B	R2.A	R2.B
a	b	a	b
a	b	x	y
x	y	a	b
x	y	x	y

5.1.4 L'union, ∪

Il existe deux autres opérateurs binaires, qui sont à la fois plus simples et moins fréquemment utilisés. Le premier est l'union. L'expression $R \cup S$ crée une relation comprenant tous les tuples existant dans l'une ou l'autre des relations R et S. Il existe une condition impérative : *les deux relations doivent avoir le même schéma*, c'est-à-dire même nombre d'attributs, mêmes noms et mêmes types.

L'union des relations $R(A, B)$ et $S(C, D)$ données en exemple ci-dessus est donc interdite (on ne saurait pas comment nommer les attributs dans le résultat). En revanche, en posant $S' = \rho_{C \to A, D \to B}(S)$, il devient possible de calculer $R \cup S'$, avec le résultat suivant :

A	B
a	b
x	y
c	d
u	v

Comme pour la projection, il faut penser à éviter les doublons. Donc le tuple `(x,y)` qui existe à la fois dans R et dans S' ne figure qu'une seule fois dans le résultat.

5.1.5 La différence, $-$

Comme l'union, la différence s'applique à deux relations qui ont le même schéma. L'expression $R - S$ a alors pour résultat tous les tuples de R qui ne sont pas dans S.

Voici la différence de R et S', les deux relations étant définies comme précédemment.

A	B
a	b

La différence est le seul opérateur qui permet d'exprimer des requêtes comportant une négation (on veut 'rejeter' quelque chose, on 'ne veut pas' des lignes ayant telle propriété). Il s'agit d'une fonctionnalité importante et difficile à manier : elle sera détaillée plus loin.

5.1.6 Jointure, $\bowtie$

Toutes les requêtes exprimables avec l'algèbre relationnelle peuvent se construire avec les 5 opérateurs présentés ci-dessus. En principe, on pourrait donc s'en contenter. En pratique, il existe d'autres opérations, très couramment utilisées, qui peuvent se contruire par composition des opérations de base. La plus importante est la jointure.

Afin de comprendre l'intérêt de cet opérateur, regardons à nouveau la Fig. 5.3 qui représente le produit cartésien `Station` $\times$ `Activite`. Le résultat est énorme et comprend manifestement un grand nombre de lignes qui ne nous intéressent pas. Cela ne présente pas beaucoup de sens de rapprocher des informations sur Santalba, aux Antilles et sur l'activité de ski à Passac.

Si, en revanche, on considère le produit cartésien comme un *résultat intermédiaire*, on voit qu'il devient maintenant possible, avec une simple sélection, de rapprocher les informations générales sur une station et la liste des activités de cette station.

La sélection est la suivante :

$$\sigma_{S.nomStation = A.nomStation}(Station \times Activite)$$

Et on obtient le résultat de la Fig. 5.4.

On a donc effectué une *composition* de deux opérations (un produit cartésien, une sélection) afin de rapprocher des informations réparties dans plusieurs tables, mais ayant des liens entre elles (toutes les informations dans un tuple du résultat sont relatives à une seule station). Cette opération est une *jointure*, que l'on peut directement, et simplement, noter :

$$Station \bowtie_{nomStation = nomStation} Activite$$

La jointure consiste donc à rapprocher les lignes de deux relations pour lesquelles les valeurs d'un (ou plusieurs) attributs sont identiques. De fait, dans 90% des cas, ces attributs communs sont (1) la clé primaire d'une des relations et (2) la clé étrangère dans l'autre relation. Dans l'exemple ci-dessus, c'est le cas pour `nomStation`. La jointure permet alors de *reconstruire* l'association entre `Station` et `Activite`.

S.nomStation	cap.	lieu	région	tarif	A.nomStation	libelle	prix
Venusa	350	Guadeloupe	Antilles	1200	Venusa	Voile	150
Venusa	350	Guadeloupe	Antilles	1200	Venusa	Plongée	120
Farniente	200	Seychelles	Océan Indien	1500	Farniente	Plongée	130
Santalba	150	Martinique	Antilles	2000	Santalba	Kayac	50
Passac	400	Alpes	Europe	1000	Passac	Ski	200
Passac	400	Alpes	Europe	1000	Passac	Piscine	20

FIG. 5.4 – *Une jointure* $\sigma_{S.nomStation=A.nomStation}(Station \times Activite)$*, composition de* $\times$ *et* σ

Une jointure $R \bowtie_F S$ peut simplement être définie comme étant équivalent à $\sigma_F(R \times S)$. Le critère de rapprochement, F, peut être n'importe quelle opération de comparaison liant un attribut de R à un attribut de S. En pratique, on emploie peu les $\neq$ ou '$<$' qui sont difficiles à interpréter, et on effectue des égalités.

Remarque : Si on n'exprime pas de critère de rapprochement, la jointure est équivalente à un produit cartésien.

Il faut être attentif aux ambiguités dans le nommage des attributs qui peut survenir dans la jointure au même titre que dans le produit cartésien. Les solutions à employer sont les mêmes : on préfixe par le nom de la relation ou par un synonyme clair, ou bien on renomme des attributs avant d'effectuer la jointure.

5.2 Expression de requêtes avec l'algèbre

Cette section est consacrée à l'expression de requêtes algébriques complexes impliquant plusieurs opérateurs. On utilise la *composition* des opérations, rendue possible par le fait que tout opérateur produit en sortie une relation sur laquelle on peut appliquer à nouveau des opérateurs.

5.2.1 Sélection généralisée

Regardons d'abord comment on peut généraliser les critères de sélection de l'opérateur σ. Jusqu'à présent on a vu comment sélectionner des lignes satisfaisant *un* critère de sélection, par exemple : 'les stations aux Antilles'. Maintenant supposons que l'on veuille retrouver les stations qui sont aux Antilles *et* dont la capacité est supérieure à 200. On peut exprimer cette requête par une composition :

$$\sigma_{capacite>200}(\sigma_{region='Antilles'}(Station))$$

Ce qui revient à pouvoir exprimer une sélection avec une *conjonction* de critères. La requête précédente est donc équivalente à celle ci-dessous, où le '$\wedge$' dénote le 'et' logique.

$$\sigma_{capacite>200 \wedge region='Antilles'}(Station)$$

Donc la composition de plusieurs sélections permet d'exprimer une conjonction de critères de recherche. De même la composition de la sélection et de l'union permet d'exprimer la *disjonction*. Voici la requête qui recherche les stations qui sont aux Antilles, *ou* dont la capacité est supérieure à 200.

$$\sigma_{capacite>200}(Station) \cup \sigma_{region='Antilles'}(Station)$$

Ce qui permet de s'autoriser la syntaxe suivante, où le '$\vee$' dénote le 'ou' logique.

$$\sigma_{capacite>200 \vee region='Antilles'}(Station)$$

Enfin la *différence* permet d'exprimer la *négation* et "d'éliminer" des lignes. Par exemple, voici la requête qui sélectionne les stations dont la capacité est supérieure à 200 mais qui ne sont *pas* aux Antilles.

$$\sigma_{capacite>200}(Station) - \sigma_{region='Antilles'}(Station)$$

Cette requête est équivalente à une sélection où on s'autorise l'opérateur '$\neq$' :

$$\sigma_{capacite>200 \wedge region\neq'Antilles'}(Station)$$

En résumé, les opérateurs d'union et de différence permettent de définir une sélection σ_F où le critère F est une expression booléenne quelconque. Attention cependant : si toute sélection avec un 'ou' peut s'exprimer par une union, l'inverse n'est pas vrai (exercice).

5.2.2 Requêtes conjonctives

Les requêtes dites *conjonctives* constituent l'essentiel des requêtes courantes. Intuitivement, il s'agit de toutes les recherches qui s'expriment avec des 'et', par opposition à celles qui impliquent des 'ou' ou des 'not'. Dans l'algèbre, ces requêtes sont toutes celles qui peuvent s'écrire avec seulement trois opérateurs : π, σ, $\times$ (et donc, indirectement, $\bowtie$).

Les plus simples sont celles où on n'utilise que π et σ. En voici quelques exemples.

1. Nom des stations aux Antilles : $\pi_{nomStation}(\sigma_{region='Antilles'}(Station))$
2. Nom des stations où l'on pratique la voile. $\pi_{nomStation}(\sigma_{libelle='Voile'}(Activite))$
3. Nom et prénom des clients européens $\pi_{nom,prenom}(\sigma_{region='Europe'}(Client))$

Des requêtes légèrement plus complexes - et extrêmement utiles - sont celles qui impliquent la jointure (le produit cartésien ne présente d'intérêt que quand il est associé à la sélection). On doit utiliser la jointure dès que les attributs nécessaires pour évaluer une requête sont réparties dans au moins deux tables. Ces "attributs nécessaires" peuvent être :

– Soit des attributs qui figurent dans le résultat ;

– Soit des attributs sur lesquels on exprime un critère de sélection.

Considérons par exemple la requête suivante : "Donner le nom et la région des stations où l'on pratique la voile". Une analyse très simple suffit pour constater que l'on a besoin des attributs `région` qui apparaît dans la relation `Station`, `libelle` qui apparaît dans `Activite`, et `nomStation` qui apparaît dans les deux relations.

Donc il faut faire une jointure, de manière à rapprocher les lignes de `Station` et de `Activité`. Il reste donc à déterminer le (ou les) attribut(s) sur lesquels se fait ce rapprochement. Ici, comme dans la plupart des cas, la jointure permet de "recalculer" l'association entre les relations `Station` et `Activité`. Elle s'effectue donc sur l'attribut `nomStation` qui permet de représenter cette association.

$$\pi_{nomStation,region}(Station \bowtie_{nomStation=nomStation} \sigma_{libelle='Voile'}(Activite))$$

En pratique, la grande majorité des opérations de jointure s'effectue sur des attributs qui sont clé primaire dans une relation, et clé secondaire dans l'autre. Il ne s'agit pas d'une règle absolue, mais elle résulte du fait que la jointure permet le plus souvent de reconstituer le lien entre des informations qui sont naturellement associées (comme une station et ses activités, ou une station et ses clients), mais qui ont été réparties dans plusieurs relations au moment de la modélisation logique de la base. Cette reconstitution s'appuie sur le mécanisme de clé étrangère qui a été étudié dans le chapitre consacré à la conception.

Voici quelques autres exemples qui illustrent cet état de fait :

– Nom des clients qui sont allés à Passac :

$$\pi_{nom}(Client \bowtie_{id=idClient} \sigma_{nomStation='Passac'}(Sejour))$$

– Quelles régions a visité le client 30 :

$$\pi_{region}(\sigma_{idClient=30}(Sejour) \bowtie_{station=nomStation} (Station))$$

– Nom des clients qui ont eu l'occasion de faire de la voile :

$$\pi_{nom}(Client \bowtie_{id=idClient} (Sejour \bowtie_{station=nomStation} \sigma_{libelle='Voile'}(Activite)))$$

La dernière requête comprend deux jointures, portant à chaque fois sur des clés primaires et/ou étrangères. Encore une fois ce sont les clés qui définissent les liens entre les relations, et elle servent donc naturellement de support à l'expression des requêtes.

Voici maintenant un exemple qui montre que cette règle n'est pas systématique. On veut exprimer la requête qui recherche les noms des clients qui sont partis en vacances dans leur région, ainsi que le nom de cette région.

Ici on a besoin des informations réparties dans les relations `Station`, `Sejour` et `Client`. Voici l'expression algébrique :

$$\pi_{nom,client.region}(Client \bowtie_{id=idClient \wedge region=region} (Sejour \bowtie_{station=nomStation} Station))$$

Les jointures avec la table `Sejour` se font sur les couples (clé primaire, clé étrangère), mais on a en plus un critère de rapprochement relatif à l'attribut `région` de `Client` et de `Station`. Noter que dans la projection finale, on utilise la notation `client.region` pour éviter toute ambiguité.

5.2.3 Requêtes avec $\cup$ et $-$

Pour finir, voici quelques exemples de requêtes impliquant les deux opérateurs $\cup$ et $-$. Leur utilisation est moins fréquente, mais elle peut s'avérer absolument nécessaire puisque ni l'un ni l'autre ne peuvent s'exprimer à l'aide des trois opérateurs "conjonctifs" étudiés précédemment. En particulier, la différence permet d'exprimer toutes les requêtes où figure une négation : on veut sélectionner des données qui *ne* satisfont *pas* telle propriété, ou tous les "untels" *sauf* les 'x' et les 'y', etc.

Illlustration concrète sur la base de données avec la requête suivante : quelles sont les stations qui *ne* proposent *pas* de voile ?

$$\pi_{nomStation}(Station) - \pi_{nomStation}(\sigma_{libelle='Voile'}(Activite))$$

Comme le suggère cet exemple, la démarche générale pour construire une requête du type 'Tous les O qui ne satisfont pas la propriété p" est la suivante :

1. Construire une première requête A qui sélectionne tous les O.
2. Construire une deuxième requête B qui sélectionne tous les O *qui satisfont* p.
3. Finalement, faire $A - B$.

Les requêtes A et B peuvent bien entendu être arbitrairement complexes et mettre en oeuvre des jointures, des sélections, etc. La seule contrainte est que le résultat de A et de B comprenne le même nombre d'attributs.

Voici quelques exemples complémentaires qui illustrent ce principe.

– Régions où il y a des clients, mais pas de station.

$$\pi_{region}(Client) - \pi_{region}(Station)$$

– Nom des stations qui n'ont pas reçu de client américain.

$$\pi_{nomStation}(Station) - \pi_{station}(Sejour \bowtie_{idClient=id} \sigma_{region='Amerique'}(Client))$$

– Id des clients qui ne sont pas allés aux Antilles.

$$\pi_{idClient}(Client) - \pi_{idClient}(\sigma_{region='Antilles'}(Station) \bowtie_{nomStation=station} Sejour)$$

La dernière requête construit l'ensemble des `idClient` pour les clients qui ne sont pas allés aux Antilles. Pour obtenir le nom de ces clients, il suffit d'ajouter une jointure (exercice).

Complément d'un ensemble

La différence peut être employée pour calculer le *complément* d'un ensemble. Prenons l'exemple suivant : on veut les ids des clients *et* les stations où ils ne sont pas allés. En d'autres termes, parmi toutes les associations Client/Station possibles, on veut justement celles qui *ne sont pas* représentées dans la base !

C'est un des rares cas où le produit cartésien seul est utile : il permet justement de constituer "toutes les associations possibles". Il reste ensuite à en soustraire celles qui sont dans la base avec l'opérateur $-$.

$$(\pi_{id}(Client) \times \pi_{nomStation}(Station)) - \pi_{idClient,station}(Sejour)$$

Quantification universelle

Enfin la différence est nécessaire pour les requêtes qui font appel à la quantification universelle : celles où l'on demande par exemple qu'une propriété soit *toujours* vraie. A priori, on ne voit pas pourquoi la différence peut être utile dans de tels cas. Cela résulte simplement de l'équivalence suivante : une propriété est vraie pour *tous* les éléments d'un ensemble si et seulement si *il n'existe pas* un élément de cet ensemble pour lequel la propriété est fausse.

En pratique, on se ramène toujours à la seconde forme pour exprimer des requêtes : donc on emploie toujours la négation et la quantification existentielle à la place de la quantification universelle.

Par exemple : quelles sont les stations dont *toutes* les activités ont un prix supérieur à 100 ? On l'exprime également par 'quelles sont stations pour lesquelles *il n'existe pas* d'activité avec un prix *inférieur* à 100'. Ce qui donne l'expression suivante :

$$\pi_{nomStation}(Station) - \pi_{nomStation}(\sigma_{prix<100}(Activite))$$

Pour finir, voici une des requêtes les plus complexes, la *division*. L'énoncé (en français) est simple, mais l'expression algébrique ne l'est pas du tout. L'exemple est le suivant : on veut les ids des clients qui sont allés dans *toutes* les stations.

Traduit avec négation et quantification existentielle, cela donne : les ids des clients tels *qu'il n'existe pas* de station où ils *ne soient pas* allés. On constate une double négation, ce qui donne l'expression algébrique suivante :

$$\pi_{id}(Client) - \pi_{id}((\pi_{id}(Client) \times \pi_{nomStation}(Station)) - \pi_{idClient,station}(Sejour))$$

On réutilise l'expression donnant les clients et les stations où ils ne sont pas allés (voir plus haut) : on obtient un ensemble B. Il reste à prendre tous les clients, sauf ceux qui sont dans B. Ce type de requête est rare (heureusement) mais illustre la capacité de l'algèbre à exprimer par de simples manipulations ensemblistes des opérations complexes.

5.3 Exercices

Exercice 8 *La figure 5.5 donne une instance de la base "Immeubles" (le schéma est légèrement simplifié).*
Pour chacune des requêtes suivantes, exprimez en français sa signification, et donnez son résultat.

1. $\pi_{nom,age}(Personne)$
2. $\pi_{nomImm}(Immeuble)$

3. $\pi_{nomImm,noApp}(\sigma_{superficie>150}(Appart))$
4. $\pi_{nomOccupant}(\sigma_{nomImm='Barabas' \vee anneeArrivee>1994}(Occupant))$
5. $\pi_{nomImm,noApp}(\sigma_{noApp=etage}(appart))$

6. $\pi_{nomGerant,superficie}(Immeuble \bowtie_{nomImm=nomImm} Appart)$
7. $\pi_{nomOccupant,anneeArrivee,superficie}(Appart \bowtie_{nomImm=nomImm \wedge noApp=noApp} Occupant)$
8. $\pi_{profession}(Immeuble \bowtie_{nomGerant=nom} Personne)$
9. $\pi_{nomGerant}(Immeuble \bowtie_{nomGerant=nomOccupant \wedge nomImm=nomImm} Occupant)$
10. $\pi_{superficie}(\sigma_{nomOccupant='Rachel'}(Occupant) \bowtie_{nomImm=nomImm \wedge noApp=noApp} Appart)$

11. $\pi_{nomImm,noApp}(Appart) - \pi_{nomImm,noApp}(Occupant)$
12. $\pi_{nomImm}(Immeuble) - \pi_{nomImm}(\sigma_{nomOccupant='Doug'}(Occupant))$.
13. $\pi_{nomImm}(\sigma_{nomOccupant\neq'Doug'}(Occupant))$.

Quelle est la différence entre les deux dernières requêtes ?

Exercice 9 *Exprimez les requêtes suivantes, en algèbre relationnelle. Pour chaque requête, donnez le résultat sur la base "Immeubles".*

1. *Nom des immeubles ayant strictement plus de 10 étages.*
2. *Nom des personnes ayant emménagé avant 1994.*
3. *Qui habite le Koudalou ?*
4. *Nom des informaticiens de plus de 25 ans.*
5. *Nom des immeubles ayant un appartement de plus de 150 m^2.*
6. *Qui gère l'appartement où habite Rachel ?*
7. *Dans quel immeuble habite un acteur ?*
8. *Qui habite un appartement de moins de 70 m^2 ?*
9. *Nom des personnes qui habitent au dernier étage de leur immeuble.*
10. *Qui a emménagé au moins 20 ans après la construction de son immeuble ?*
11. *Profession du gérant du Barabas ?*
12. *Couples de personnes ayant emménagé dans le même immeuble la même année.*
13. *Age et profession des occupants de l'immeuble géré par Ross ?*
14. *Qui habite, dans un immeuble de plus de 10 étages, un appartement de plus de 100 m^2 ?*
15. *Couples de personnes habitant, dans le même immeuble, un appartement de même superficie.*
16. *Qui n'habite pas un appartement géré par Ross ?*
17. *Qui n'habite pas un appartement qu'il gère lui-même ?*

nomImm	adresse	nbEtages	annéeConstruction	nomGérant
Koudalou	3 Rue Blanche	15	1975	Doug
Barabas	2 Allee Nikos	2	1973	Ross

La table *Immeuble*

nomImm	noApp	superficie	étage
Koudalou	1	150	14
Koudalou	34	50	15
Koudalou	51	200	2
Koudalou	52	50	5
Barabas	1	250	1
Barabas	2	250	2

La table *Appart*

nom	âge	profession
Ross	51	Informaticien
Alice	34	Cadre
Rachel	23	Stagiaire
William	52	Acteur
Doug	34	Rentier

La table *Personne*

nomImm	noApp	nomOcc	anneeArrivee
Koudalou	1	Rachel	1992
Barabas	1	Doug	1994
Barabas	2	Ross	1994
Koudalou	51	William	1996
Koudalou	34	Alice	1993

La table *Occupant*

FIG. 5.5 – *Les immeubles et leurs occupants*

18. *Quels sont les immeubles où personne n'a emménagé en 1996 ?*
19. *Quels sont les immeubles où tout le monde a emménagé en 1994 ?*

Exercice 10

1. *Montrez que la requête 4 dans l'exercice 8 peut s'exprimer avec l'union.*
2. *Inversement, donner une requête sur la base 'Immeuble' qui peut s'exprimer avec l'union, mais pas avec σ et $\vee$.*
3. *Exprimez la dernière requête de l'exercice 8 avec la différence, et sans $\neq$.*