Licences Mathématiques et Informatique
3ème année - Formation Initiale et par Apprentissage

Bases de données relationnelles

Polycopié de cours - Algèbre relationnelle

Maude Manouvrier

Table des matières

Chapitre 3

Algèbre relationnelle

L'algèbre relationnelle est un langage de requête procédural [19] : on dit comment obtenir un résultat ensembliste. Il est composé d'un ensemble d'Opérateurs qui prennent en entrée une ou deux instances de relations (i.e. un ou 2 ensembles de nuplets) et offrent en sortie une nouvelle instance relation.

Notation : On notera en majuscules les noms de schéma de relation (R,S etc.) et on notera en minuscules le instances de relations (r, s, ...).

3.1 Opérateurs unaires fondamentaux

3.1.1 Opérateur de sélection

L'opérateur de sélection sélectionne dans l'instance des nuplets qui satisfont un certain prédicat. La sélection est noté σ. le prédicat apparaît en indice du symbole σ. L'argument de l'opérateur est donné entre parenthèses.

Par exemple, si on suppose que l'on a une relation `Etudiant(EtudiantID, Nom, Prenom, ...Ville, ...NbAnnees)`, pour sélectionner les étudiants habitants Paris, la requête est :

$$\sigma_{(Ville=''paris'')}(Etudiant)$$

La relation résultat a le même schéma que la relation `Etudiant`.

Les symboles de comparaison d'un prédicat de sélection sont : $=$, $\neq$, $>$, $<$, $\leq$, $\geq$. Il est possible de combiner plusieurs prédicats en utilisant les symboles et ($\wedge$) et ou ($\vee$).

Par exemple, pour sélectionner les étudiants habitant Paris et ayant passé au moins deux ans à l'université :

$$\sigma_{(Ville=''paris'')\wedge(NbAnnees\geq 2)}(Etudiant)$$

Il est à noter que n'importe quelle comparaison avec une valeur nulle est évaluée à faux.

Un exemple est donné sur les transparents 50 et 51.

3.1.2 Opérateur de projection

L'opérateur de projection permet de créer une relation en éliminant certains attributs d'une instance de relation. Les lignes dupliquées sont éliminées. La projection est représentée par la lettre Π. Les attributs devant apparaître dans la relation résultat sont indiqués en indice du Π. L'ensemble de nuplets (instance) argument apparaît entre parenthèses. La relation résultat a pour attribut les attribut en indice de Π.

Par exemple, pour créer une relation contenant tous les prénoms et les noms des étudiants, on écrit :

$$\Pi_{Nom,Prenom}(Etudiant)$$

La relation résultat a pour schéma (`Nom,Prenom`).

Un exemple est donné sur les transparents 50 et 51.

3.1.3 Combinaison des opérateurs

Il est possible de combiner les opérateurs. Par exemple, pour obtenir les noms et prénoms des étudiants parisiens, on écrit :

$$\Pi_{Nom,Prenom}(\sigma_{(Ville=''Paris)}(Etudiant))$$

Il faut faire attention à l'ordre des opérateurs. La requête suivante n'est pas valide :

$$\sigma_{(Ville=''Paris)}\,[\Pi_{Nom,Prenom}(Etudiant)]$$

Un exemple est présenté sur le transparent 50.

3.2 Opérateurs binaires fondamentaux

3.2.1 Union de deux relations

L'union de deux instances de relations permet de rassembler les nuplets de deux instances au sein d'une seule. Pour que l'union $r \cup s$ soit valide, il faut que les schémas des relations soient **union-compatibles**, c'est-à-dire qu'il faut vérifier les deux conditions suivantes :

1. les relations r et s ont le même nombre d'attributs (les noms des attributs n'importent pas).
2. le domaine du $i$ème attribut de r et celui du $i$ème attributs de s sont les mêmes, quel que soit i.

$r \cup s$ a le même schéma que R et contient les nuplets de r et les nuplets de s (sans doublons).

Par exemple, si on suppose que l'on a une relation `Enseignant(EnseignantID, Nom, Prenom, Grade, #DepartementID)`, si l'on souhaite unir les noms et prénoms des étudiants et des enseignants, on écrit :

$$\Pi_{Nom,Prenom}(Etudiant) \cup \Pi_{Nom,Prenom}(Enseignant)$$

Attention, le SGBD ne réfléchit pas. Il exécute uniquement ce qu'on lui demande. Par conséquent, il serait possible de faire la requête suivante, même si elle n'a aucun sens (le type des attributs étant chaîne de caractères pour tous les attributs) :

$$\Pi_{Nom,Prenom}(Etudiant) \cup \Pi_{Rue,Ville}(Enseignant)$$

Un exemple est présenté sur les transparents 53, 58 et 59.

3.2.2 Différence de deux relations

La différence de deux instances de relations, notée $-$, permet de savoir quels sont les nuplets appartenant à une instance et n'appartenant pas à une autre. L'expression $r - s$ correspond à la relation, de schéma R, contenant les nuplets de r qui n'apparaissent pas dans s. Il faut que les schémas des relations soient union-compatibles.

Par exemple si l'on souhaite les noms et prénoms des étudiants qui ne sont pas enseignants :

$$\Pi_{Nom,Prenom}(Etudiant) - \Pi_{Nom,Prenom}(Enseignant)$$

Un exemple est présenté sur les transparents 54, 58 et 59.

3.2.3 Produit-cartésien

Le produit cartésien, noté $*$ ou $\times$ permet de combiner les informations de deux instances de relations. Il n'y a pas de contraintes sur les schémas de des relations. Le schéma de $r * s$, aussi noté $r \times s$, est RS, i.e. tous les attributs de R, suivis de tous les attributs de S, en répétant les attributs de même nom.

Si on suppose que l'on a une relation `Departement(`<u>`DepartementID`</u>`, NomDepartement)`, le produit cartésien des relations `Enseignant` et `Departement` a pour schéma :
`(EnseignantID, Nom, Prenom, Grade, Enseignant.DepartmentID, Department.`<u>`DepartmentID`</u>`,`
`NomDepartment)`

La relation résultat du produit cartésien des relations `Enseignant` et `Departement` associe chaque nuplet de la relation `Enseignant` à un nuplet de la relation `Departement`.
Des exemples sont donnés sur les transparents 55, 60 et 61.

De manière plus formelle : Si on a les relations r_1 de shcéma R_1 et r_2 de schéma R_2, alors le produit cartésien $r = r_1 * r_2$ a pour schéma la concaténation de R_1 et R_2. La relation r contient tous les nuplets t tels qu'il existe un nuplet t_1 dans r_1 et un nuplet t_2 dans r_2 pour lesquels $\Pi_{R_1}(t) = t_1$ et $\Pi_{R_2}(t) = t_2$.

Là encore, il est possible de combiner les opérateurs. Par exemple, si l'on souhaite les noms et prénoms des professeurs associés au nom du département auquel ils appartiennent :

$$\Pi_{Nom,Prenom,NomDepartement}[\sigma_{Departement.DepartmentID=Enseignant.DepartmentID}(Enseignant*Departement)]$$

3.2.4 Renommage

Il est possible de renommer un attribut en utilisant le symbole $\longrightarrow$. Le renommage peut-être utilisé pour faciliter l'écriture de certaines requêtes.
Par exemple : $\Pi_{LastName,FirstName}(Enseignant_{Nom\longrightarrow LastName,Prenom\longrightarrow FirstName})$
Un exemple est présenté sur le transparent 57.

3.3 Autres opérateurs

Les opérateurs fondamentaux sont suffisants pour exprimer n'importe quelle requête en algèbre relationnelle. Certains opérateurs ont été ajoutés pour simplifier l'expression (l'écriture) de certaines requêtes.

3.3.1 Intersection de deux relations

L'intersection de deux instances de relations r et s (avec $R \cap S \neq \emptyset$) est définie telle que :

$$r \cap s = r - (r - s)$$

Un exemple est présenté sur les transparents 57, 58 et 59.

3.3.2 Theta-jointure

La thêta-jointure, notée $\bowtie_\theta$ avec θ un prédicat, est définie par :

$$r \bowtie_\theta s = \sigma_\theta(r * s)$$

Cet opérateur permet de joindre (associer) les nuplets de 2 instances vérifiant certaines conditions θ.

Des exemples sont présentés sur les transparents 63 et 64.

3.3.3 Jointure naturelle

La jointure naturelle de deux relations r et s telles que $R \cap S = \{A_1, A_2, ..., A_n\}$ est définie par :

$$r \bowtie s = \Pi_{R \cup S}(\sigma_{(r.A_1 = s.A_1) \wedge (r.A_2 = s.A_2) \wedge ... \wedge (r.A_n = s.A_n)}(r * s)$$

Cet opérateur permet de joindre (associer) les nuplets de 2 instances qui ont la même valeur sur les attributs de même nom dans les 2 relations.

Un exemple de jointure naturelle est présentée sur les transparents 60 et 62.

3.3.4 Division

La division, notée $\div$, correspond à une requête qui contient en général le terme "pour tous".

Soient r et s deux instances de relations avec $S \subseteq R$ (chaque attribut de S est dans R). La relation $r \div s$ est une relation de schéma $R - S$, c'est-à-dire contenant les attributs de R n'apparaissant pas dans S. Un nuplet t appartient à la relation $r \div s$ si et seulement si il vérifie les deux conditions suivantes :

1. t appartient à $\Pi_{R-S}(r)$
2. Pour tout nuplet t_s de s, il existe un nuplet t_r dans r qui satisfait les deux conditions suivantes :
   (a) $\Pi_S t_r = t_s$
   (b) $\Pi_{R-S} t_r = t$

Cette opérateur signifie que les nuplets t résultats sont associés dans r à tous les nuplets de s. $r \div s$ contient donc les morceaux de nuplets de r, de schéma $(R - S)$ qui, dans r, sont associés à tous les nuplets de s. En définitive :

$$r \div s = \Pi_{R-S}(r) - \Pi_{R-S}[(\Pi_{R-S}(r) * s) - \Pi_{R-S,S}(r)]$$

En effet, $\Pi_{R-S}(r)$ donne tous les nuplets qui satisfont la première condition.

$\Pi_{R-S}(r) * s$ associe tous les nuplets de Π_{R-S} à ceux de s.

$\Pi_{R-S,S}(r)$ ne fait que réordonner les attributs de r pour permettre de faire correctement la différence.

D'où : $(\Pi_{R-S}(r)*s)-\Pi_{R-S,S}(r)$ donne les paires d'attributs de $\Pi_{R-S}(r)$ et s qui n'apparaissent pas dans r.

Soit t_j un nuplet de $\Pi_{R-S}[(\Pi_{R-S}(r) * s) - \Pi_{R-S,S}(r)]$ alors il existe un nuplet t_s de s qui ne se combine pas avec t_j pour former un nuplet de r. Par conséquent, t_j contient les valeurs des attributs de $(R - S)$ qui n'apparaissent pas dans $r \div s$. Ces valeurs sont éliminées de $\Pi_{R-S}(r)$.

Prenons un exemple simple. Soit une relation `R(Etudiant,Cours)` dont une instance est donnée ci-dessous.

TABLE 3.1 – Une instance r de la relation `R(Etudiant,Cours)`

Etudiant	**Cours**
Toto	Proba
Titi	Proba
Toto	Stats
Titi	BD
Toto	BD
Tutu	BD

On veut connaître le noms des étudiants qui assistent à tous les cours. La liste des cours peut être obtenue en faisant la requête $s = \Pi_{Cours}(r)$:

TABLE 3.2 – Le résultat de la requête $s = \Pi_{Cours}(r)$

Cours
Proba
Stats
BD

Pour trouver les étudiants assistant à tous les cours, il faut donc récupérer les valeurs de l'attribut `Etudiant` qui, dans la relation r, sont associées à toutes les nuplets de s (dans l'exemple $Toto$).

Vérifions la formule donnée précédemment, avec les relations $R(Etudiant, Cours)$ et $S(Cours)$. On a :

1. La liste des étudiants : $\Pi_{R-S}(r) = \Pi_{Etudiant}(r)$.

TABLE 3.3 – Le résultat de la requête $\Pi_{R-S}(r) = \Pi_{Etudiant}(r)$

Etudiant
Toto
Titi
Tutu

2. L'association de chaque étudiant à chaque cours qu'il le suive ou non :
   $\Pi_{Etudiant}(r) * s$

TABLE 3.4 – Le résultat de la requête $\Pi_{R-S}(r) * s$

Etudiant	**Cours**
Toto	Proba
Toto	Stats
Toto	BD
Titi	Proba
Titi	Stats
Titi	BD
Tutu	Proba
Tutu	Stats
Tutu	BD

3. On enlève $\Pi_{R-S,S}(r)$ (qui dans cet exemple est égale à $\Pi_{Etudiant,Cours}(r) = r$) : on a donc les étudiants associé à un cours auquel ils n'ont pas assisté (qui ne nous intéressent pas pour le résultat de la requête) : $\Pi_{Etudiant}[(\Pi_{Etudiant}(r) * s) - \Pi_{Etudiant,Cours}(r)]$

TABLE 3.5 – Le résultat de la requête $\Pi_{Etudiant}[(\Pi_{Etudiant}(r) * s) - \Pi_{Etudiant,Cours}(r)]$ (la différence entre les nuplets de la Table 3.4 et ceux de la Table 3.1)

Etudiant	**Cours**
Titi	Stats
Tutu	Proba
Tutu	Stats

4. Le résultat final de la requête est donc :
$\Pi_{Etudiant}(r) - \Pi_{Etudiant}[(\Pi_{Etudiant}(r) * s) - \Pi_{Etudiant,Cours}(r)]$. Il reste bien le nuplet $(Toto)$.

Un autre exemple de division est donné sur les transparents 66 à 70 (sur le transparent 70, un enseignement est identifié par 2 attributs, le numéro de l'enseignement et celui du département).

3.4 Contraintes référentielles

L'algèbre relationnelle permet d'exprimer certaines contraintes, en particulier les contraintes d'intégrité référentielles [21].

Un exemple de contrainte référentielle est : *"Tous les numéros de Département apparaissant dans la relation Enseignant doivent être contenus dans la relation Departement"*. Ce qui signifie que les valeurs prises par l'attribut $DepartmentID$ dans la relation $Enseignant$ doivent apparaître dans les valeurs prises par l'attribut $DepartmentID$ dans la relation $Departement$: $DepartmentID$ est une clé étrangère dans la relation $Enseignant$. Cette contrainte s'exprime en algèbre relationnelle de la manière suivante :

$$\Pi_{DepartmentID}(Enseignant) \subseteq \Pi_{DepartmentID}(Departement)$$

Si en plus il est spécifié que *" tous les départements doivent contenir au moins un enseignant"*, alors on peut exprimé la contrainte par :

$$\Pi_{DepartmentID}(Departement) - \Pi_{DepartmentID}(Enseignant) = \emptyset$$

3.5 Projection généralisée

Il est possible d'étendre la projection en ajoutant des fonctions arithmétiques. La projection généralisée a pour forme :

$$\Pi_{F_1,F_2,\ldots,F_n}(r)$$

où r est une expression en algèbre relationnelle et chaque F_i sont des expressions arithmétiques s'appliquant sur les constantes ou les attributs du schéma de r. Une expression arithmétique peut être une constante seule ou un attribut seul.

Par exemple si on a une relation `Compte(Num_Client, Nom_Client,Debit, Credit)` on peut demander le solde du compte de chaque client :

$$\Pi_{Nom\_Client,(Credit-Debit)}(Compte)$$

Un exemple est donné sur le transparent 90.

3.5.1 Jointure externe (*outer-join*)

La jointure externe est une extension de l'opérateur de jointure qui permet de conserver les informations manquantes (i.e. les nuplets ne participant pas à la jointure).
Il est possible de faire :

- une jointure externe à droite (*right outer-join*), notée $\bowtie\!\sqsubset$,
- une jointure externe à gauche (*left outer-join*), notée $\sqsupset\!\bowtie$,
- où une jointure externe entière (*full outer-join*), notée $\sqsupset\!\bowtie\!\sqsubset$.

$r \sqsupset\!\bowtie s$ effectue la jointure entre r et s et conserve toutes les valeurs des attributs des nuplets de r (la relation de gauche) qui ne joignent avec rien, les valeurs attributs des nuplets de s étant remplacés par des valeurs nulles. La jointure externe à droite est l'Opérateur symétrique. La jointure externe entière traite les deux cas à la fois.
Par exemple (repris de [19]) :

TABLE 3.6 – Une instance de la relation *Personne*.

Nom_Employé	**Adresse**	**Ville**
Tom	rue machin	Marseille
Jerry	rue bidule	Paris
Alex	rue truc	Limoges
Marthe	rue machin	Perpignan

TABLE 3.7 – Une instance de la relation *Employé*.

Nom_Employé	**Nom_Filiale**	**Salaire**
Tom	SUD_EST	10000
Jerry	IDF	25000
Sophie	IDF	15000
Marthe	SUD_OUEST	12000

TABLE 3.8 – Le résultat de $Personne$ ⟕ *Employé*.

Nom_Employé	**Adresse**	**Ville**	**Nom_Filiale**	**Salaire**
Tom	rue machin	Marseille	SUd_EST	10000
Jerry	rue bidule	Paris	IDF	25000
Marthe	rue machin	Perpignan	SUD_OUEST	12000
Alex	rue truc	Limoges	NULL	NULL

3.5.2 Fonctions d'agrégation

Les fonctions d'agrégation prennent en paramètres une collection de valeurs et retournent une seule valeur résultat. Ces opérateurs sont :

- **Sum** : qui permet de faire la somme de valeurs (entières ou réelles).
- **Avg** : qui calcule la moyennes de valeurs (entières ou réelles.
- **Count** : qui retourne le nombre d'éléments dans la collection.
- **Min** : qui retourne la plus petite valeur de la collection.
- **Max** : qui retourne la plus grande valeur de la collection.
- **Count_distinct** : qui est identique à *count* mais ne tient pas compte des doublons.

Il est possible d'appliquer des opérateurs d'agrégation non pas sur des ensembles de nuplets, mais sur des groupes d'ensembles de nuplets. Par exemple, on veut connaître le nombre d'enseignants par Département :

$$Group_{Nom\_Departement, count_{EnseignantID}} (Enseignant \bowtie Departement)$$

L'expression $Group_{Nom\_Departement}$ signifie que la relation résultat de la jointure entre *Enseignant* et *Departement* doit être divisée en groupes basés sur l'identifiant du département :

Le résultat de la requête $Group_{Nom\_Departement, count_{EnseignantID}} (Enseignant \bowtie Departement)$ donne :

TABLE 3.9 – Le résultat de $Personne$ ⟖ *Employé*.

Nom_Employé	**Adresse**	**Ville**	**Nom_Filiale**	**Salaire**
Tom	rue machin	Marseille	SUd_EST	10000
Jerry	rue bidule	Paris	IDF	25000
Marthe	rue machin	Perpignan	SUD_OUEST	12000
Sophie	NULL	NULL	IDF	15000

TABLE 3.10 – Le résultat de *Personne* $\sqsupset\!\bowtie\!\sqsubset$ *Employé*.

Nom_Employé	**Adresse**	**Ville**	**Nom_Filiale**	**Salaire**
Tom	rue machin	Marseille	SUd_EST	10000
Jerry	rue bidule	Paris	IDF	25000
Marthe	rue machin	Perpignan	SUD_OUEST	12000
Alex	rue truc	Limoges	NULL	NULL
Sophie	NULL	NULL	IDF	15000

TABLE 3.11 – Le regroupement des enseignants par département.

EnseignantID	**DepartmentID**	**Nom**	...	**Nom_Departement**
1	2	Toto	...	Math
3	2	Tales	...	Math
2	1	Manou	...	Info

TABLE 3.12 – Le nombre d'enseignants par département.

Department_Name	**count**
Math	2
Info	1

Bibliographie

[1] D. Austin, *Using Oracle8^{TM}*, Simple Solutions - Essential Skills, QUE, 1998, ISBN : 0-7897-1653-4

[2] R. Chapuis, *Oracle 8*, Editions Dunes et Laser, 1998, ISBN : 2-913010-07-5

[3] P. Chen, *The Entity-Relationship Model–Toward a Unified View of Data*, ACM Transactions on Database Systems, Vol. 1, No. 1, March 1976, Pages 9-36, http ://www.csc.lsu.edu/~chen/pdf/erd.pdf

[4] T. Connolly, C. Begg et A. Strachan, *Database Systems - A pratical Approach to Design, Implementation and Management*, Addison-Wesley, 1998, ISBN : 0-201-34287-1, disponible à la BU 055.7 CON

[5] C.J. Date, *Introduction aux bases de données*, 6ème édition, Thomson Publishing, 1998, ISBN : 2-84180-964-1, disponible à la BU 005.7 DAT

[6] R. Elamsri et S.B. Navathe, *Fundamentals of Database Systems*, 3ème édition, Addison Wesley- disponible à la BU 005.7 ELM

[7] P. Delmal, *SQL2 - Application à Oracle, Access et RDB*, 2ème édition, Bibliothèque des Universités - Informatique, De Boeck Université, 1988, ISBN : 2-8041-2995-0,disponible à la BU 005.74 SQL

[8] S. Feuerstein, B. Pribyl et C. Dawes, *Oracle PL/SQL - précis et concis*, O'Reilly, 2000, ISBN : 2-84177-108-3

[9] G. Gardarin, *Bases de Données - objet & relationnel*, Eyrolles, 1999, ISBN : 2-212-09060-9, disponible à la BU 005.74 GAR

[10] R. Grin, *Introduction aux bases de données, modèle relationnel*, Université Sophia-Antipolis, jan. 2000

[11] R. Grin, *Langage SQL*, Université Sophia-Antipolis, jan. 2000

[12] G. Gardarin et O. Gardarin *Le Client-Serveur*, Eyrolles, 1999, disponible à la BU

[13] H. Garcia-Molina, J.D. Ulmann et J. Widow, *Database SYstem Implementation*, Prentice Hall, 2000, ISBN :0-13-040264-8, disponible à la BU 005.7 GAR

[14] H. Garcia-Molina, J.D. Ulmann et J. Widow, *Database Systems - The Complete Book*, Prentice Hall, 2002, ISBN :0-13-031995-3

[15] S. Krakowiak, *Gestion Physique des données*, Ecole Thématique "Jeunes Chercheurs" en Base de Données, Volume 2, Port Camargue, mars 1997

[16] D. Lockman, *Oracle8^{TM} Dévéloppement de bases de données*, Le programmeur - Formation en 21 jours, Editions Simon et Schuster Macmillan (S&SM), 1997, ISBN : 2-7440-0350-6, disponible à la BU 005.74 ORA

[17] P.J. Pratt, *Initiation à SQL - Cours et exercices corrigés*, Eyrolles, 2001, ISBN : 2-212-09285-7

[18] R. Ramakrishnan et J. Gehrke, *Database Management Systems*, Second Edition ; McGraw-Hill, 2000, ISBN : 0-07-232206-3, disponible à la BU 055.7 RAM

[19] A. Silberschatz, H.F. Korth et S. Sudarshan, *Database System Concepts*, Third Edition, McGraw-Hill, 1996, ISBN : 0-07-114810-8, disponible à la BU 005.7 DAT

[20] C. Soutou, *De UML à SQL - Conception de bases de données*, Eyrolles, 2002, ISBN : 2-212-11098-7

[21] J.D. Ullman et J. Widom, *A first Course in Database Systems*, Prentice Hall, 1997, ISBN : 0-13-887647-9, disponible à la BU 005.7 ULL