Algèbre relationnelle

L'algèbre relationnelle

L'algèbre relationnelle utilise une collection d'opérateurs qui agissent sur des relations et produisent des relations comme résultat.

Opérateurs :

- unaires : projection, sélection
- ensemblistes : union, produit, différence, intersection
- binaires (ou n-aires) : jointure, division

L'algèbre relationnelle

La projection

La projection d'une relation R sur un ensemble d'attributs $\{A_i\}$ de R est un sous-ensemble de R réduit aux sous-tuples ne contenant que les attributs de $\{A_i\}$.

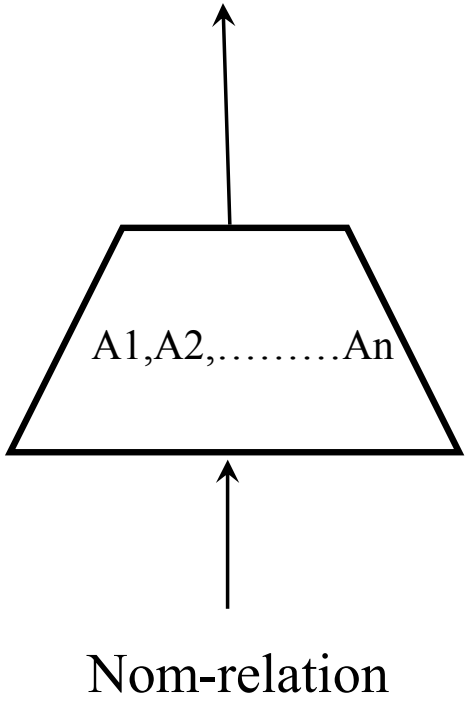
Suppression dans les tuples de tous les attributs n'appartenant pas à $\{A_i\}$, puis suppression des doublons.

Notation : $\Pi_{A_1,\ldots,A_n}$(nom-relation)

L’algèbre relationnelle

La projection

Représentation graphique



L'algèbre relationnelle

La projection

$\Pi_{code,adresse}$ (Étudiant)

Étudiant

Code	Adresse
00134	Adr1
00134	Adr1
01045	Adr2
01045	Adr2

L'algèbre relationnelle

La projection

$$\Pi_{code,adresse}\ (Étudiant)$$

Code	Adresse
00134	Adr1
00134	Adr1
01045	Adr2
01045	Adr2

Code	Adresse
00134	Adr1
01045	Adr2

L'algèbre relationnelle

La sélection

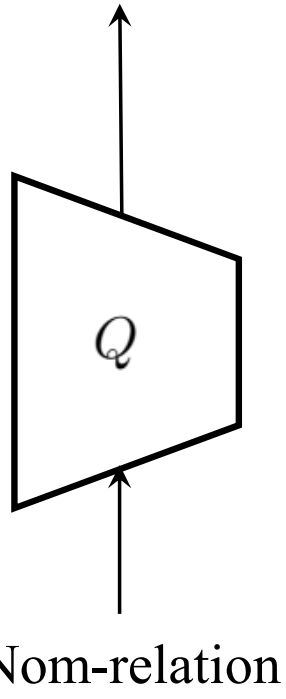
La sélection sur une relation R suivant une qualification Q portant sur des attributs de R est un sous-ensemble de R dont les tuples satisfont Q.

Notation : σ_Q(nom-relation)

L’algèbre relationnelle

La sélection

Représentation graphique



L'algèbre relationnelle

La sélection

Étudiant

Code	Nom	prénom	adresse
00134	Nom1	Pr1	Adr1
01045	Nom2	Pr2	Adr2
00033	Nom3	Pr3	Adr3
01190	Nom4	Pr4	Adr4

L'algèbre relationnelle

La sélection

σ Nom = 'Nom2' (Étudiant)

Code	Nom	prénom	adresse
00134	Nom1	Pr1	Adr1
01045	Nom2	Pr2	Adr2
00033	Nom3	Pr3	Adr3
01190	Nom4	Pr4	Adr4

L'algèbre relationnelle

La sélection

σ Nom = 'Nom2' (Étudiant)

Code	Nom	prénom	adresse
01045	Nom2	Pr2	Adr2

L'algèbre relationnelle

L'union, U

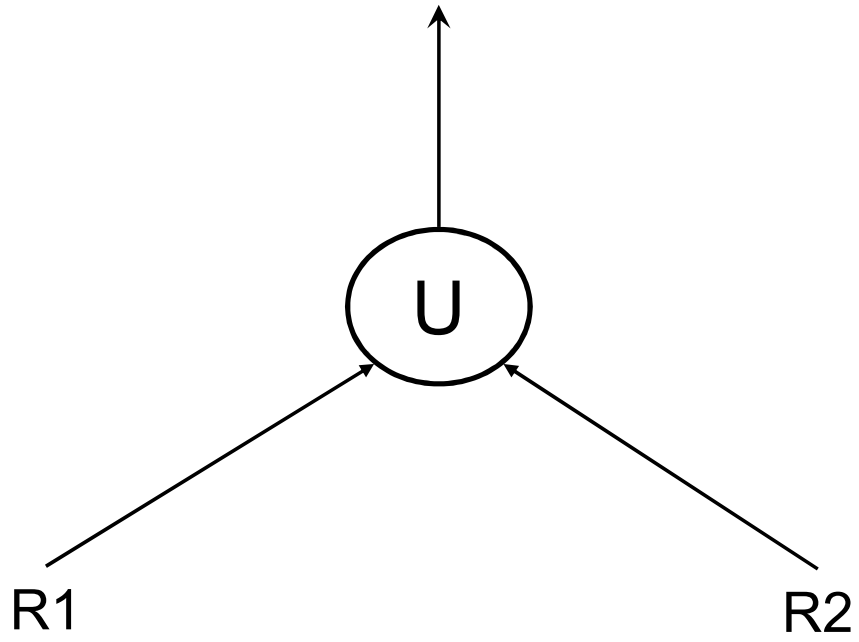
L'union est une opération ensembliste portant sur 2 relations R_1 et R_2 de même schéma. $R_1 \cup R_2$ est la relation de même schéma contenant les tuples de R_1 et ceux de R_2.

Copie des tuples de R_1 et de R_2, puis suppression des doublons.

L’algèbre relationnelle

L’union

Représentation graphique



L'algèbre relationnelle

L'union

Étudiant 1

Code	Nom	prénom	adresse
00134	Nom1	Pr1	Adr1
01045	Nom2	Pr2	Adr2
00033	Nom3	Pr3	Adr3

U

Étudiant 2

Code	Nom	prénom	adresse
00134	Nom1	Pr1	Adr1
01046	Nom4	Pr4	Adr4

L'algèbre relationnelle

L'union

Étudiant 1 ∪ Étudiant 2

Code	Nom	prénom	adresse
00134	Nom1	Pr1	Adr1
01045	Nom2	Pr2	Adr2
00033	Nom3	Pr3	Adr3
01046	Nom4	Pr4	Adr4

L'algèbre relationnelle

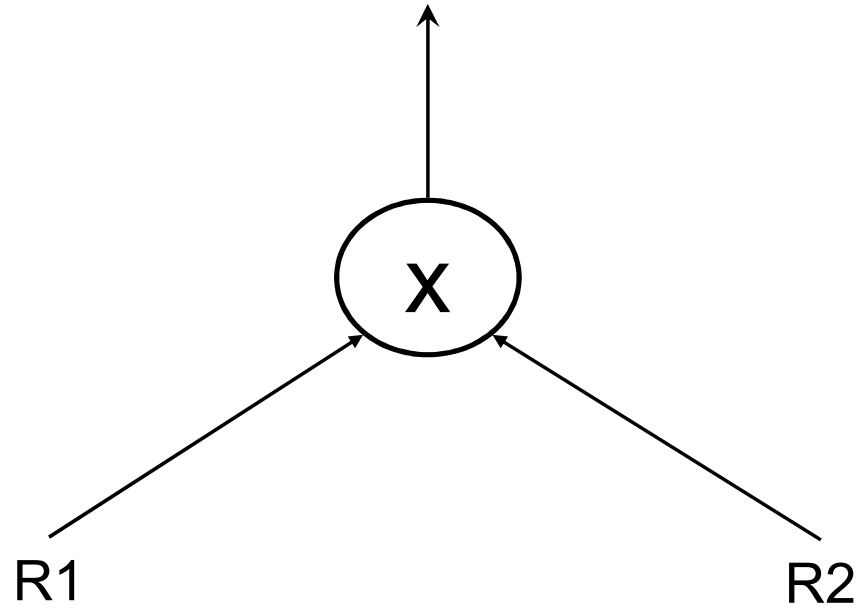
Le produit

Le produit est une opération ensembliste portant sur 2 relations R_1 et R_2. $R_1 \times R_2$ est la relation ayant pour schéma la concaténation des schémas de R_1 et R_2, et pour tuples toutes les combinaisons de tuples de R_1 et R_2.

L’algèbre relationnelle

Le produit

Représentation graphique



L'algèbre relationnelle

Le produit

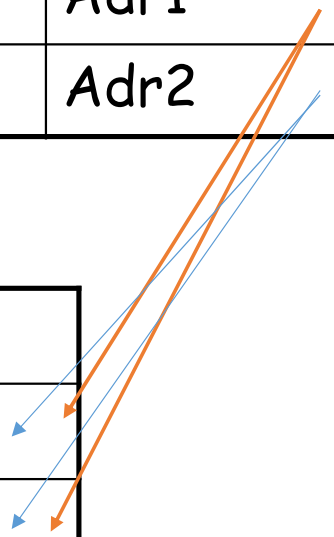
Étudiant

Code	Nom	prénom	adresse
00134	Nom1	Pr1	Adr1
01045	Nom2	Pr2	Adr2

X

Année

Année	Moyenne
2001	Bonne
2004	Passable



L'algèbre relationnelle

Le produit Étudiant X Année

Code	Nom	Prénom	Adresse	Année	Moyenne
00134	Nom1	Pr1	Adr1	2001	Bonne
00134	Nom1	Pr1	Adr1	2004	Passable
01045	Nom2	Pr2	Adr2	2001	Bonne
01045	Nom2	Pr2	Adr2	2004	Passable

L'algèbre relationnelle

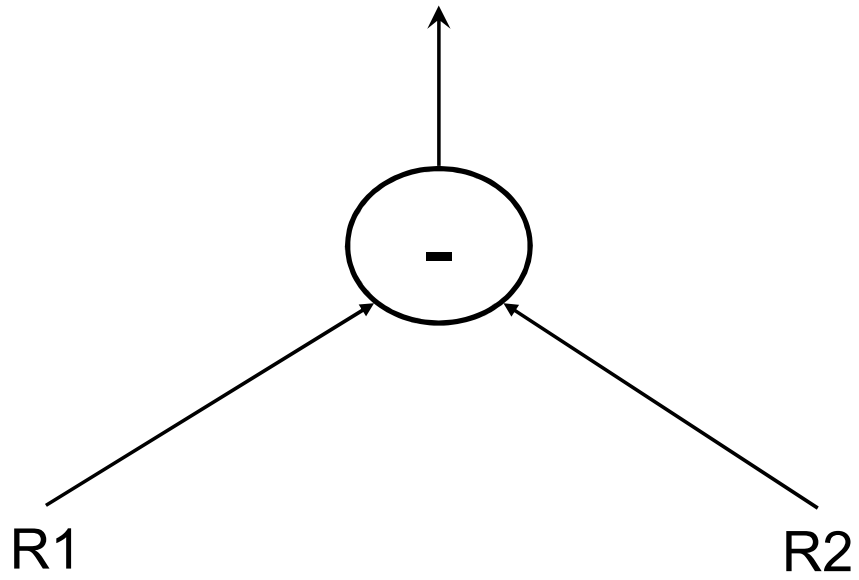
La différence

La différence est une opération ensembliste portant sur 2 relations R_1 et R_2 de même schéma. $R_1 - R_2$ est la relation ayant le même schéma que R_1 et R_2, et pour tuples ceux appartenant à R_1 et pas à R_2.

L’algèbre relationnelle

La différence

Représentation graphique



L'algèbre relationnelle

La différence

Étudiant 1

Code	Nom	prénom	adresse
00134	Nom1	Pr1	Adr1
01045	Nom2	Pr2	Adr2
00033	Nom3	Pr3	Adr3

-

Étudiant 2

Code	Nom	prénom	adresse
00134	Nom1	Pr1	Adr1
01046	Nom4	Pr4	Adr4

L'algèbre relationnelle

La différence

Étudiant 1 - Étudiant 2

Code	Nom	prénom	adresse
01045	Nom2	Pr2	Adr2
00033	Nom3	Pr3	Adr3

L'algèbre relationnelle

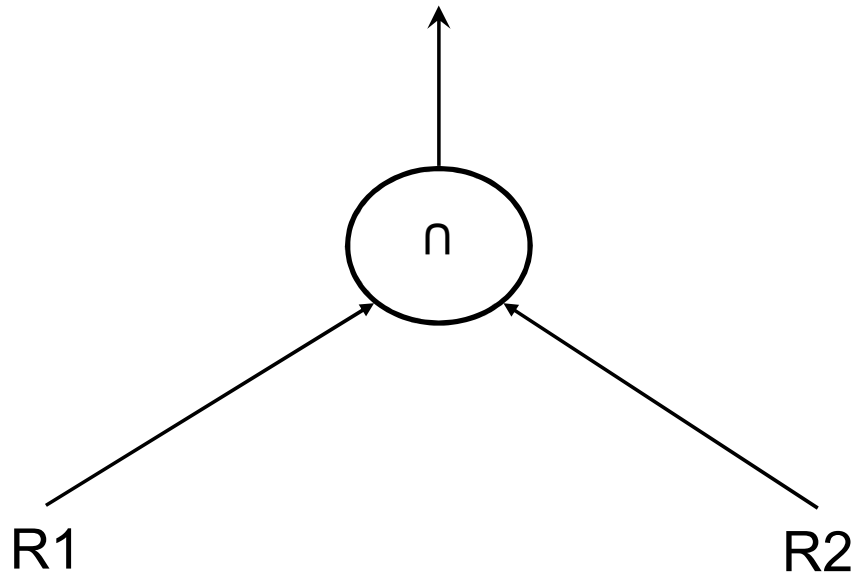
L'intersection

L'intersection est une opération ensembliste portant sur 2 relations R_1 et R_2 de même schéma. $R_1 \cap R_2$ est la relation ayant le même schéma que R_1 et R_2, et pour tuples ceux appartenant à la fois à R_1 et à R_2.

L’algèbre relationnelle

L’intersection

Représentation graphique



L'algèbre relationnelle

L'intersection

Étudiant 1

Code	Nom	prénom	adresse
00134	Nom1	Pr1	Adr1
01045	Nom2	Pr2	Adr2
00033	Nom3	Pr3	Adr3

∩

Étudiant 2

Code	Nom	prénom	adresse
00134	Nom1	Pr1	Adr1
01046	Nom4	Pr4	Adr4

L'algèbre relationnelle

L'intersection

Étudiant 1 ∩ Étudiant 2

Code	Nom	prénom	adresse
00134	Nom1	Pr1	Adr1

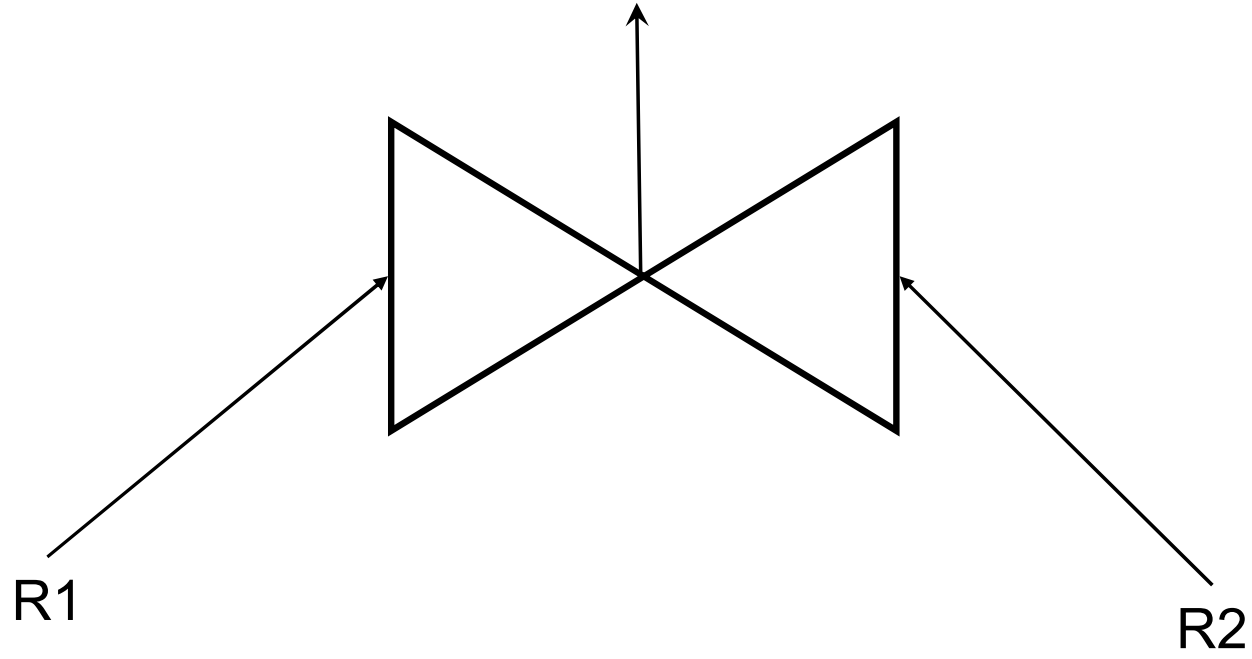
L'algèbre relationnelle

La Jointure

La jointure est une opération ensembliste portant sur 2 relations R_1 et R_2. $R_1 \bowtie_{\text{attributs}} R_2$ est la relation dont les attributs sont ceux de R_1 et ceux de R_2, et dont les tuples sont obtenus en composant un tuple de R_1 et un tuple de R_2 ayant la même valeur pour les attributs précisés.

L’algèbre relationnelle

La jointure



$$\text{Jointure} = \sigma_{\text{condition}}(R1 \times R2)$$

L'algèbre relationnelle

La jointure

Étudiant $\underset{code}{\bowtie}$ Prêt

Étudiant

Code	Nom	Prénom	Adresse
0134	Nom1	Pr1	Adr1
0178	Nom2	Pr2	Adr2
1045	Nom3	Pr3	Adr3
1196	Nom4	Pr4	Adr4

Prêt

Code	Cod-livre	Date
0178	MA12	12-03-2015
1182	IN08	12-03-2015

L'algèbre relationnelle

La jointure

$$\text{Étudiant} \underset{\text{code}}{\bowtie} \text{Prêt}$$

Code	Nom	Prénom	Adresse	Cod-livre	Date
0178	Nom2	Pr2	Adr2	MA12	12-03-2015

SQL (Structured Query Language)

SQL

- langage relationnel permettant de :
  - définir et modifier le schéma d'une base de données relationnelle
  - interroger et modifier une base de données relationnelle
  - contrôler la sécurité et l'intégrité de la base
- langage interactif ou intégré dans un langage de programmation

SQL

SQL est un langage relationnel ⇒ on manipule des tables et on obtient des tables

Une instruction SQL = une requête

3 familles d'opérations :

- LDD = Langage de Définition des Données : description de la structure de la base de données (tables, attributs)
- LMD = Langage de Manipulation de Données : manipulation des tables
- LCD = Langage de Contrôle des Données : gestion du contrôle et de la sécurité de la base de données

SQL

- Création d'une Table

1) **CREATE TABLE** Banque (Code_Agence CHAR(50),Ville CHAR(20) NOT NULL DEFAULT 'Nom_Ville', PRIMARY KEY (Code_Agence)) ;

2) **CREATE TABLE** Etudiant (Matricule CHAR(9) ,……..);

CREATE TABLE Module (Code_Mod CHAR(6),….....);

CREATE TABLE Etude_Mod (
Matricule CHAR(9),
Code_Mod CHAR(6),
Note Integer **CHECK (Note BETWEEN O and 20)**,
Decision CHAR(8) **CHECK (Decision IN ('Admis', 'Ajourné'))**,
PRIMARY KEY (Matricule , Cod_Mod),
FOREIGN KEY (Matricule) REFERENCES Etudiant,
FOREIGN KEY (Code_Mod) REFERENCES Module) ;

- Suppression d'une Table

DROP TABLE Banque;

SQL

- **Insertion de tuples**

INSERT INTO <NomTable> (A1, A2, ... An) VALUES (v1, v2, ... vn);

Exemple:

INSERT INTO Étudiant (code, nom, prénom, adresse) values (0456,nom1, pr1,adr1);

SQL

- **Modification de tuples**

UPDATE <NomTable>
SET A1=v1, A2=v2, ... An=vn
WHERE condition;

Exemple :
UPDATE Étudiant
SET adresse=‘Alger’
WHERE code=‘0234’;

SQL

- **Suppression de tuples**

DELETE FROM <NomTable>
WHERE condition;

Exemple :
DELETE FROM Étudiant
WHERE code=‘0234’;

SQL

SELECT liste colonnes (projetées, calculées)
FROM listes tables
WHERE prédicat de jointure AND prédicat de sélection;

Exemple1
SELECT *
FROM Produit
WHERE codeP='P653';

Exemple2

SELECT Num_Emp , Nom_Emp
FROM EMPLOYES , DEPARTEMENTS
WHERE (EMPLOYES•Num_Dept=DEPARTEMENTS•Num_Dept) **AND (**Ville = ‘ANNABA');

SQL

SELECT liste colonnes (projetées, calculées)
FROM listes tables
[WHERE prédicat de jointure AND prédicat de sélection]
[GROUP BY colonnes de groupement]
[HAVING prédicat de sélection de groupes]
[ORDER BY critères et ordre de tri];

SQL

- Prédicat de sélection composé par:

 Comparateurs arithmétiques (<, <=, >, >=, != (<>), =)
 Connecteurs logiques (AND, OR, NOT)
 Comparateur de chaines LIKE
 Intervalle BETWEEN
 Liste IN
 Valeurs nulles IS NULL
 Requête imbriquée IN, EXISTS

SQL

SQL offre aussi un certain nombre de fonctions dites fonctions de groupes qui peuvent être utilisées dans l'expression d'une requête. Les plus importantes sont :

- **AVG** : permet de calculer une **MOYENNE**
- **SUM** : permet de calculer une **SOMME**
- **COUNT** : permet de compter des tuples
- **MAX** : permet de calculer un **maximum**
- **MIN** : permet de calculer un **minimum**
- **GROUP BY** : permet de créer des sous-ensembles de tuples
- **HAVING** : permet de tester si une condition est vérifiée par un groupe de tuples

SQL

- **Tri du résultat d'une requête : la clause ORDER BY**

La clause ORDER BY permet de trier le résultat retourné par une requête. L'ordre peut être ascendant (ASC) ou descendant (DESC)

- **Exemple1**

Q: Liste des employés du département 10 par ordre décroissant de leur salaire?

SELECT Nom_Emp
FROM EMPLOYES
WHERE Num_Dept = 10
ORDER BY Salaire **DESC;**

SQL

Exemple2 :

```
SELECT region, COUNT(nomStation)
FROM Station
GROUP BY region
HAVING COUNT(nomStation) >= 3;
```