Algèbre relationnelle

Dr N. BAME

Interrogation d'une BD relationnelle

- **Langages algébriques :**
  - Une requête est une **composition d'opérations** : **algèbre relationnelle**
- **Langages logiques :**
  - Une requête est exprimée par une formule logique : **calcul relationnel**
- **Et SQL ?**
  - **sémantique** formelle = calcul relationnel
  - **algorithmes d'évaluation sont fondés** sur **l'algèbre relationnelle**

Algèbre relationnelle

- **Langage théorique**, avec des **opérations** qui permettent de **manipuler des relations** (tables)

- ***Ensemble d'opérations*** qui s'applique ***sur*** des ***relations*** pour donner de ***nouvelles relations***

- Ces opérations s'effectuent grâce à un certains nombres ***d'opérateurs***

- On distingue les opérateurs ***ensemblistes*** issues de la théorie des ensembles et les opérateurs purement ***relationnels***

Algèbre relationnelle

- **Opérateurs unaires :**

  <Opérateur>$_{\textbf{<parametres>}}$ **<Opérande>**

  → <Résultat>

- **Opérateurs binaires :**

  <Opérande><Opérateur>$_{\textbf{<parametres>}}$**<Opérande>**

  → <Résultat>

- **Langage fermé :** les **opérandes** et les **résultats** sont toujours des **relations**

Les opérateurs de l'AR

- **Opérateurs de base :**
  - **Projection**
  - **Sélection**
  - **Produit cartésien**
  - opérations ensemblistes: **union**, **différence**

- **Opérateurs dérivés :**
  - **Intersection**
  - **Jointure**

Opérateurs de l'AR

- **Projection (π)** : création d'un **schéma sous-ensemble** d'un **autre schéma**. La projection élimine les n-uplets doublons.

$$\Pi_{A_1,\ldots,A_n}(R)=\{t.A_1\ldots A_n \mid t\in R\}$$

- **Sélection ou restriction (σ): Même schéma** mais **réduction du nombre de tuples** grâce à **un critères** (ou prédicat)

$$\sigma_F(R)=\{t \mid t\in R \text{ et } F(t) \text{ est vrai}\}$$

Opérateurs de l'AR

- ***Produit de cartésien (×)*** : Création d'un schéma à partir de 2 tables avec **concaténation des attributs** et **combinaison** systématique des tuples

$$R \times S = \{t.A_1 \ldots A_{k_1}\ A_{k_1+1} \ldots A_{k_1+k_2} \mid t.A_1 \ldots A_{k_1} \in R\ et\ t.A_{k_1+1} \ldots A_{k_1+k_2} \in S\}$$

- ***Jointure (⋈)*** : Création d'un table à partir de 2 tables avec ***union des attributs*** et ***concaténation des tuples*** satisfaisant à la ***condition*** de jointure

$$R \bowtie_F S = \{t.A_1 \ldots A_n B_1 \ldots B_m \mid t.A_1 \ldots A_n \in R \text{ et } t.B_1 \ldots B_m \in S \text{ et } F \text{ est vrai}\}$$

Opérateurs de l'AR

- ***Union (u)*** **:** création d'une table a partir de **2 tables** ayant le ***même schéma***. La table résultant contient ***l'ensemble*** des tuples

$$R \cup S = \{t \mid t \in R \text{ ou } t \in S\}$$

- ***Différence (-)*** : création d'une table à partir de **2 tables** ayant le ***même schéma***. La table résultante contient les ***tuples appartenant à une*** table et pas à l'autre

$$R - S = \{t \mid t \in R \text{ et } t \notin S\}$$

- ***Intersection (∩):*** création d'une table des tuples ***communs*** de 2 tables ayant le **même *schéma***

$$R \cap S = \{t \mid t \in R \text{ et } t \in S\}$$

Restriction

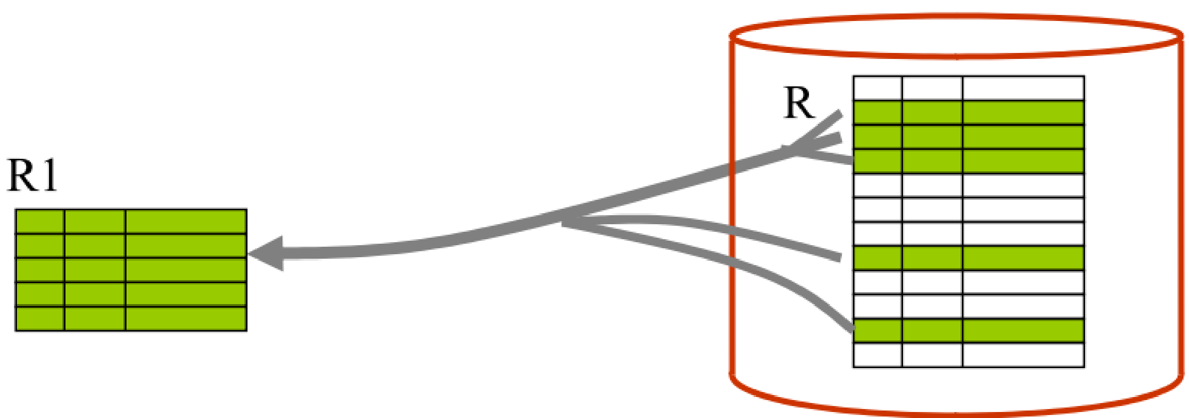
Sélection d'un sous-ensemble de la relation opérande:

$$\sigma_{prédicat}(R)=\{t \mid t \in R \text{ et } \textbf{prédicat(t)} \text{ est } \textbf{vrai}\}$$

Où

- **R** est une relation, *t* est un n-uplet variable
- ***Prédicat*** est une ***formule logique (condition)*** composée de :
  - opérandes: constantes et attributs
  - opérateurs de comparaison :**<, >, =, ≠, ≤, ≥**
  - opérateurs logiques : **∧, ∨, ¬**

Le **résultat** est **un extrait de la relation** qui correspond aux **tuples qui satisfont le prédicat** de sélection



Exemple de sélection

EMP

ENO	ENAME	TITLE
E1	J. Doe	Elect. Eng.
E2	M. Smith	Syst. Anal.
E3	A. Lee	Mech. Eng.
E4	J. Miller	Programmer
E5	B. Casey	Syst. Anal.
E6	L. Chu	Elect. Eng.
E7	R. Davis	Mech. Eng.
E8	J. Jones	Syst. Anal.

Les employés de profession ‘Elect. Eng’

$\sigma_{\text{TITLE='Elect. Eng.'}}(\text{EMP})$

ENO	ENAME	TITLE
E1	J. Doe	Elect. Eng
E6	L. Chu	Elect. Eng.

Projection

Projection sur un ensemble d'attributs d'une relation

$$\Pi_{A1,\ldots,An}(R)=\{t.A_1 \ldots A_n \mid t \in R\}$$

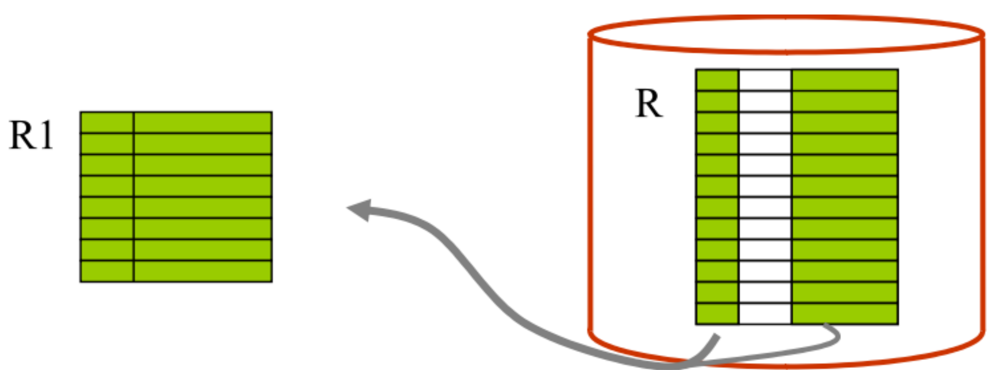
Où

- R est une relation, t est une variable n-uplet
- $\{A_1 ,\ldots, A_n\}$ est un sous-ensemble des attributs de R
- **Le résultat ne contient que les attributs spécifiés**

Note: la projection *élimine les n-uplets doublons*.

Les SGBD relationnels (et SQL) permettent

- projection (pure)
- projection sans élimination des doublons



Exemple de projection

PROJ

PNO	PNAME	BUDGET
P1	Instrumentation	150000
P2	Database Develop.	135000
P3	CAD/CAM	250000
P4	Maintenance	310000
P5	CAD/CAM	500000

$\Pi_{PNO,BUDGET}(PROJ)$

PNO	BUDGET
P1	150000
P2	135000
P3	250000
P4	310000
P5	500000

$\Pi_{PNAME}(PROJ)$

PNAME
Instrumentation
Database Develop.
CAD/CAM
Maintenance

Exercices

- Soit le schéma relationnel ci-dessous :

  Emp (Eno, Ename, #Title, City)

  Pay(Title, Salary)

  Project(Pno, Pname, Budget, City)

  Works(#Eno, #Pno, Resp, Dur)

- Exprimer les requêtes suivantes en algèbre relationnelle :

1. Villes où sont localisés des projets?
2. Villes où il y a des employés?
3. Projets de budget > 225?
4. Projets de Dakar ?
5. Employés informaticiens ?
6. Budgets des projets de Thies ?
7. Noms et budgets des projets de Thies ?
8. Villes où habitent des informaticiens ?
9. Noms des professions de salaires supérieurs à 1.000.000 ?

Produit cartésien

Produit cartésien entre deux tables :

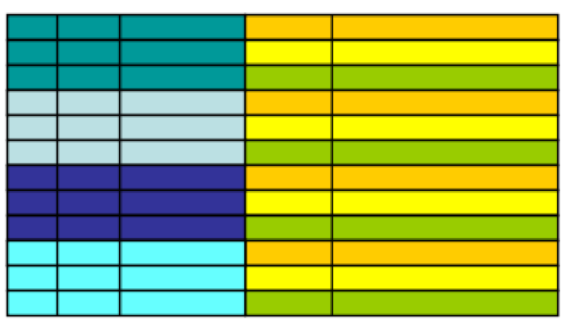
$R1 \times R2 = \{t.A_1 \ldots A_{k1}\ A_{k1+1} \ldots A_{k1+k2} \mid t.A_1 \ldots A_{k1} \in R1$ et $t.A_{k1+1} \ldots A_{k1+k2} \in R2\}$

Ou

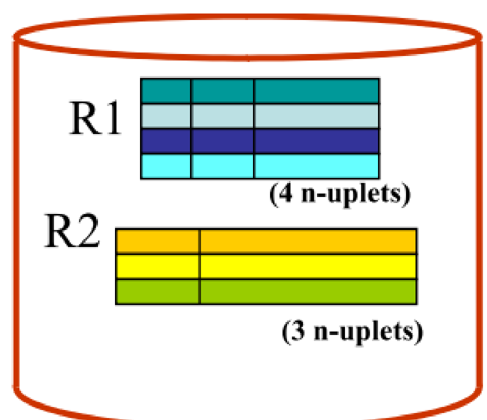
- R1 est une table de **degré** (nombre de colonnes) $\boldsymbol{k_1}$, **cardinalité** (nombres de n-uplets) $\boldsymbol{n_1}$
- R2 est une table de**degré** (nombre de colonnes) $\boldsymbol{k_2}$, **cardinalité** (nombres de n-uplets) $\boldsymbol{n_2}$

Le résultat est une relation de **degré ($k_1 + k_2$)** et contient **($n_1 * n_2$) n-uplets**, où chaque n-uplet est la concaténation d'un n-uplet de R1 avec un n-uplet de R2.

R



(3x4 = 12 n-uplets)



Exemple de produit cartésien

EMP

ENO	ENAME	TITLE
E1	J. Doe	Elect. Eng
E2	M. Smith	Syst. Anal.
E3	A. Lee	Mech. Eng.
E4	J. Miller	Programmer
E5	B. Casey	Syst. Anal.
E6	L. Chu	Elect. Eng.
E7	R. Davis	Mech. Eng.
E8	J. Jones	Syst. Anal.

PAY

TITLE	SALARY
Elect. Eng.	55000
Syst. Anal.	70000
Mech. Eng.	45000
Programmer	60000

EMP × PAY

ENO	ENAME	EMP.TITLE	PAY.TITLE	SALARY
E1	J. Doe	Elect. Eng.	Elect. Eng.	55000
E1	J. Doe	Elect. Eng.	Syst. Anal.	70000
E1	J. Doe	Elect. Eng.	Mech. Eng.	45000
E1	J. Doe	Elect. Eng.	Programmer	60000
E2	M. Smith	Syst. Anal.	Elect. Eng.	55000
E2	M. Smith	Syst. Anal.	Syst. Anal.	70000
E2	M. Smith	Syst. Anal.	Mech. Eng.	45000
E2	M. Smith	Syst. Anal.	Programmer	60000
E3	A. Lee	Mech. Eng.	Elect. Eng.	55000
E3	A. Lee	Mech. Eng.	Syst. Anal.	70000
E3	A. Lee	Mech. Eng.	Mech. Eng.	45000
E3	A. Lee	Mech. Eng.	Programmer	60000
E8	J. Jones	Syst. Anal.	Elect. Eng.	55000
E8	J. Jones	Syst. Anal.	Syst. Anal.	70000
E8	J. Jones	Syst. Anal.	Mech. Eng.	45000
E8	J. Jones	Syst. Anal.	Programmer	60000

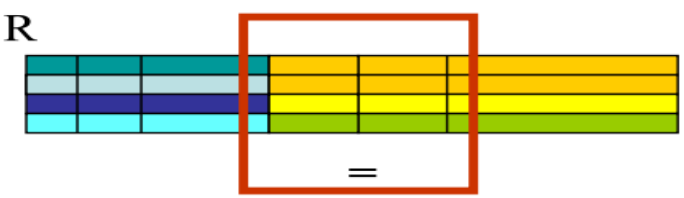
Jointure

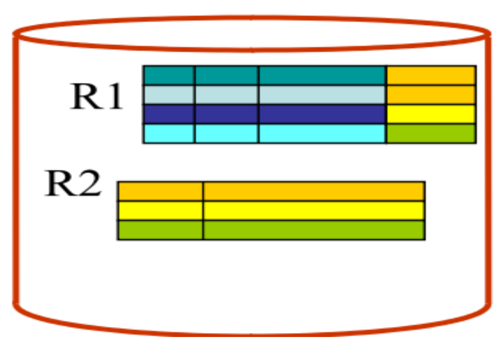
Le **résultat** R contient **les combinaisons** des n-uplets de **R1** avec les n-uplets de **R2** qui vérifient le ***prédicat de jointure***.

$R1 \otimes_F R2 = \{t.A_1 \ldots A_n\, B_1 \ldots B_m \mid t.A_1 \ldots A_n \in R1$ et $t.B_1 \ldots B_m \in R2$ et F est vrai$\}$

Où

- R1 et R2 sont des relations
- t est une variable n-uplet F est une formule logique composée de **condition** de la forme $\mathbf{A_i\ \theta\ B_j}$ où $\theta \in \{<, >, =, \neq, \leq, \geq\}$.
- La jointure s'exprime aussi par un produit cartésien suivi d'une sélection: $R1 \otimes_F R2$ $\sigma_F(R1 \times R2)$





Exemple de jointure

EMP

ENO	ENAME	TITLE
E1	J. Doe	Elect. Eng
E2	M. Smith	Syst. Anal.
E3	A. Lee	Mech. Eng.
E4	J. Miller	Programmer
E5	B. Casey	Syst. Anal.
E6	L. Chu	Elect. Eng.
E7	R. Davis	Mech. Eng.
E8	J. Jones	Syst. Anal.

WORKS

ENO	PNO	RESP	DUR
E1	P1	Manager	12
E2	P1	Analyst	24
E2	P2	Analyst	6
E3	P3	Consultant	10
E3	P4	Engineer	48
E4	P2	Programmer	18
E5	P2	Manager	24
E6	P4	Manager	48
E7	P3	Engineer	36
E7	P5	Engineer	23
E8	P3	Manager	40

$EMP \bowtie_{EMP.ENO>WORKS.ENO} WORKS$

EMP. ENO.	ENAME	TITLE	WORKS. ENO	PNO	RESP	DUR
E2	M. Smith	Syst. Anal.	E1	P1	Manager	12
E3	A. Lee	Mech. Eng.	E1	P1	Manager	12
E3	A. Lee	Mech. Eng.	E2	P1	Analyst	24
E3	A. Lee	Mech. Eng.	E2	P2	Analyst	6
E4	J. Miller	Programmer	E1	P1	Manager	12
E4	J. Miller	Programmer	E2	P1	Analyst	24
E4	J. Miller	Programmer	E2	P2	Analyst	6
E4	J. Miller	Programmer	E3	P3	Consultant	10
E4	J. Miller	Programmer	E3	P4	Engineer	48
E5	B. Casey	Syst. Anal.	E1	P1	Manager	12
E5	B. Casey	Syst. Anal.	E2	P1	Analyst	24
E5	B. Casey	Syst. Anal.	E2	P2	Analyst	6
E5	B. Casey	Syst. Anal.	E3	P3	Consultant	10
E5	B. Casey	Syst. Anal.	E3	P4	Engineer	48
E5	B. Casey	Syst. Anal.	E4	P2	Programmer	18
E6	L. Chu	Elect. Eng.	E1	P1	Manager	12
E6	L. Chu	Elect. Eng.	E2	P1	Analyst	24
E6	L. Chu	Elect. Eng.	E2	P2	Analyst	6
E6	L. Chu	Elect. Eng.	E3	P3	Consultant	10
E6	L. Chu	Elect. Eng.	E3	P4	Engineer	48
E6	L. Chu	Elect. Eng.	E4	P2	Programmer	18
E6	L. Chu	Elect. Eng.	E5	P2	Manager	24
…	…	…	…	…	…	…

Types de jointure

θ-jointure

– la formule **F** utilise les comparaisons **<, >, ≠, ≤, ≥**

Equi-jointure

– la formule F n'utilise que **l'égalité** : **=** $R \bowtie_{R.A=S.B} S$

Jointure naturelle : R(X, Y1), S(X, Y2)

- **Equi-jointure** où on **élimine les attributs en communs**

$$\boxed{R \bowtie S} = \Pi_{R.X,\, R.Y,\, S.Y'} \sigma_F (R \times S) = \Pi_{S.X,\, R.Y,\, S.Y'} \sigma_F (R \times S)$$

- la condition de jointure F est **R.X = S.X** (X représente tous **les attributs en commun** entre R et S)

Exemple de jointure naturelle

EMP

ENO	ENAME	TITLE
E1	J. Doe	Elect. Eng
E2	M. Smith	Syst. Anal.
E3	A. Lee	Mech. Eng.
E4	J. Miller	Programmer
E5	B. Casey	Syst. Anal.
E6	L. Chu	Elect. Eng.
E7	R. Davis	Mech. Eng.
E8	J. Jones	Syst. Anal.

PAY

TITLE	SALARY
Elect. Eng.	55000
Syst. Anal.	70000
Mech. Eng.	45000
Programmer	60000

EMP⋈PAY

ENO	ENAME	TITLE	SALARY
E1	J. Doe	Elect. Eng.	55000
E2	M. Smith	Analyst	70000
E3	A. Lee	Mech. Eng.	45000
E4	J. Miller	Programmer	60000
E5	B. Casey	Syst. Anal.	70000
E6	L. Chu	Elect. Eng.	55000
E7	R. Davis	Mech. Eng.	45000
E8	J. Jones	Syst. Anal.	70000

Exercices

- Soit le schéma relationnel ci-dessous :

  Emp (Eno, Ename, #Title, City)
  Pay(Title, Salary)
  Project(Pno, Pname, Budget, City)
  Works(#Eno, #Pno, Resp, Dur)

- Exprimer les requêtes suivantes en algèbre relationnelle :

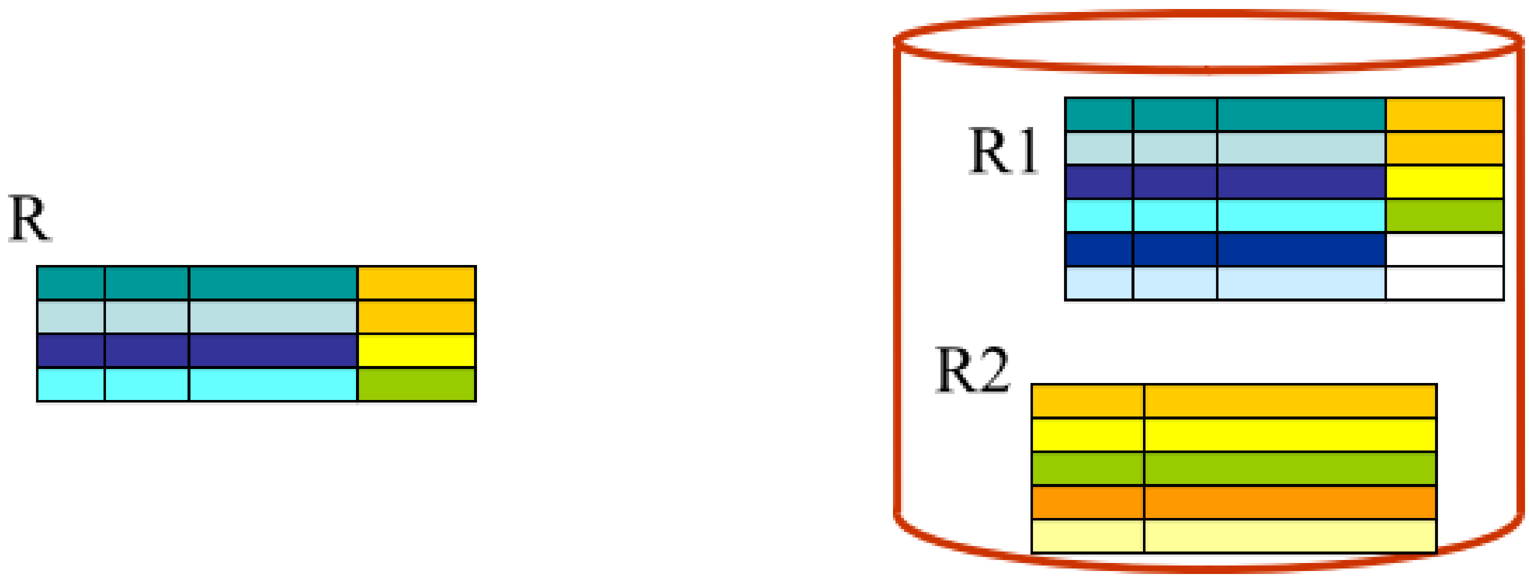
1. Noms et salaires des employés de Dakar ?
2. Noms et villes des employés ayant un salaire supérieur à 700.000 ?
3. Noms et budgets des projets où a travaillé l'employé de numéro 4 ?
4. Noms et professions des employés qui ont travaillé dans un projet pendant moins de 6 mois ?
5. Responsabilités occupées par des informaticiens dans des projets ?
6. Noms, budgets et ville des projets où a travaillé l'employé Alpha Diallo?
7. Noms, budgets et villes des projets où ont travaillé des employés ayant un salaire supérieurs à 1.200.000 ?

Semi-jointure

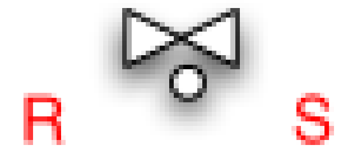
R = SEMI JOINTURE (R1, R2, <Prédicat_de_Jointure>)

R a le même schéma que R1
R contient les combinaisons des n-uplets de R1 avec les n-uplets de R2 qui vérifient le prédicat de jointure.

R =R1 $\ltimes$ R2



Jointure Externe



la **jointure externe** de 2 tables R et S est une table T obtenue par jointure de R et S et **ajout** des tuples de R et de S ne **participant pas à la jointure** avec des **valeurs nulles** pour les attributs de l'autre table

On distingue :

- la **jointure externe *droite*** ⟖
  - elle garde seulement les tuples sans correspondant de la table de droite
- la **jointure externe *gauche*** ⟕
  - elle garde seulement les tuples sans correspondant de la table de gauche

Exemple de jointure externe

VINS	Cru	Millésime	Qualité
	Volnay	1983	A
	Volnay	1979	B
	Julienas	1986	C

LIEU	Cru	Région	QualMoy
	Volnay	Bourgogne	A
	Chablis	Bourgogne	A
	Chablis	Californie	B

VINS-LIEU = VINS $\underset{\circ}{\bowtie}$ LIEU :

VINS-LIEU	Cru	Millésime	Qualité	Région	QualMoy
	Volnay	1983	A	Bourgogne	A
	Volnay	1979	B	Bourgogne	A
	Chablis	-	-	Bourgogne	A
	Chablis	-	-	Californie	B
	Julienas	1986	C	-	-

Conflit de noms

- 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**.
- **Exemple**: La table T(A, B) a les mêmes noms d'attributs que R(A, B).
- Le résultat du produit cartésien RXT a pour schéma **(A,B,A,B)**

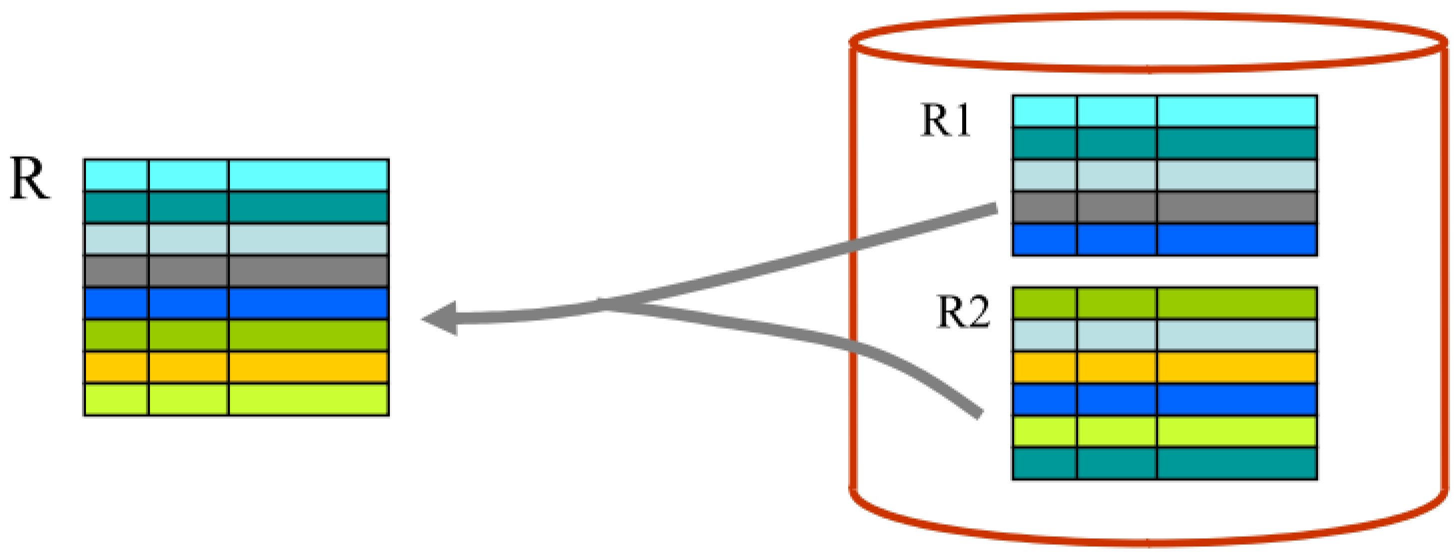
Solution

1. La première solution pour lever l'ambiguité est de **préfixer** un attribut **par le nom de la table** d'où il provient.
2. le renommage. Il s'agit d'un opérateur particulier, **dénoté ρ** qui permet de **renommer** un ou plusieurs attributs d'une relation.

$$\rho_{A\text{->}C,\ B\text{->}D}(T)$$

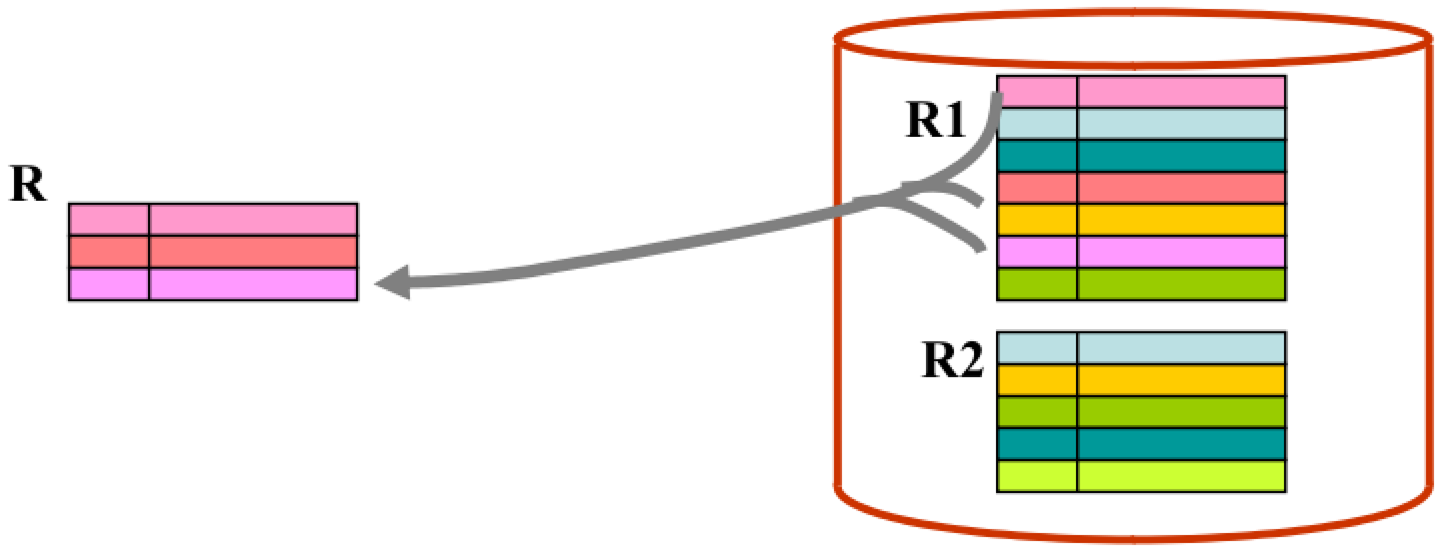
Union

- ***L'union*** *R U 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.



Différence

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



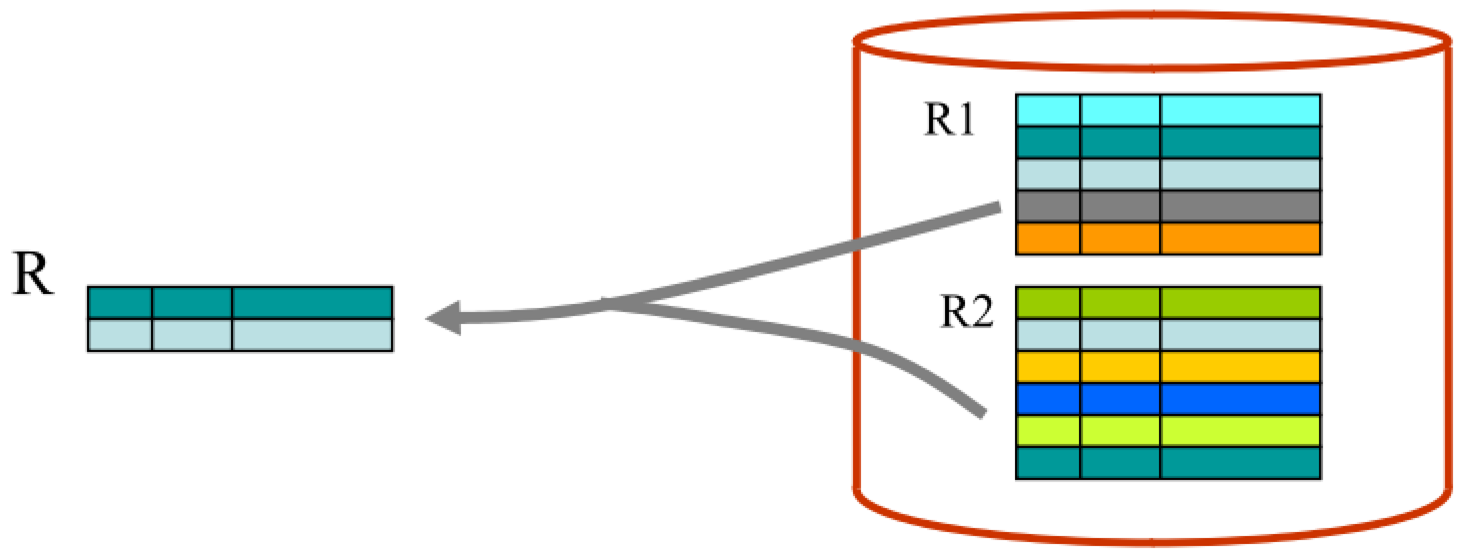
Intersection

- Intersection de deux tables:

  $R \cap S = \{t \mid t \in R \text{ et } t \in S\}$

où

- R, S sont deux tables compatibles pour l'intersection (même schéma)



Utilisation des opérateurs

- Dans les requêtes seuls les opérateurs les plus maniables sont utilisés : ce sont ***l'union*** et la ***différence*** pour *l'insertion* et la *suppression* de tuples dans la base et la ***restriction***, la ***projection*** et la ***jointure*** pour la *recherche sélective* de tuples.

- Les **opérateurs de l'algèbre relationnelle sont** à la **base** des **langages** de manipulations de données (**SQL**)

Requêtes algébriques

Emp (Eno, Ename, Title, City)

Pay(<u>Title</u>, Salary)

Project(<u>Pno</u>, Pname, Budget, City)

Works(<u>Eno</u>, <u>Pno</u>, Resp, Dur)

- Villes où il y a des employés ou des projets?

- Villes où il y a des projets mais pas d'employés?

Requêtes algébriques

Emp (Eno, Ename, Title, City)

Pay(<u>Title</u>, Salary)

Project(<u>Pno</u>, Pname, Budget, City)

Works(<u>Eno</u>, <u>Pno</u>, Resp, Dur)

- Villes où il y a des employés ou des projets?

  Π_{City} (Emp) $\cup$ Π_{City} (Project)

- Villes où il y a des projets mais pas d'employés?

  Π_{City}(Project) – Π_{City}(Emp)

Requêtes algébriques

Emp (Eno, Ename, Title, City) **Project**(Pno, Pname, Budget, City)

Pay(Title, Salary) **Works**(Eno, Pno, Resp, Dur)

Noms des projets de budget > 225?

Noms et budgets des projets où travaille l'employé E1?

Requêtes algébriques

Emp (Eno, Ename, Title, City) **Project**(Pno, Pname, Budget, City)

Pay(Title, Salary) **Works**(Eno, Pno, Resp, Dur)

Noms des projets de budget > 225?

$$\Pi_{Pnam}e(\sigma_{Budget>225}\ (Project))$$

Noms et budgets des projets où travaille l'employé E1?

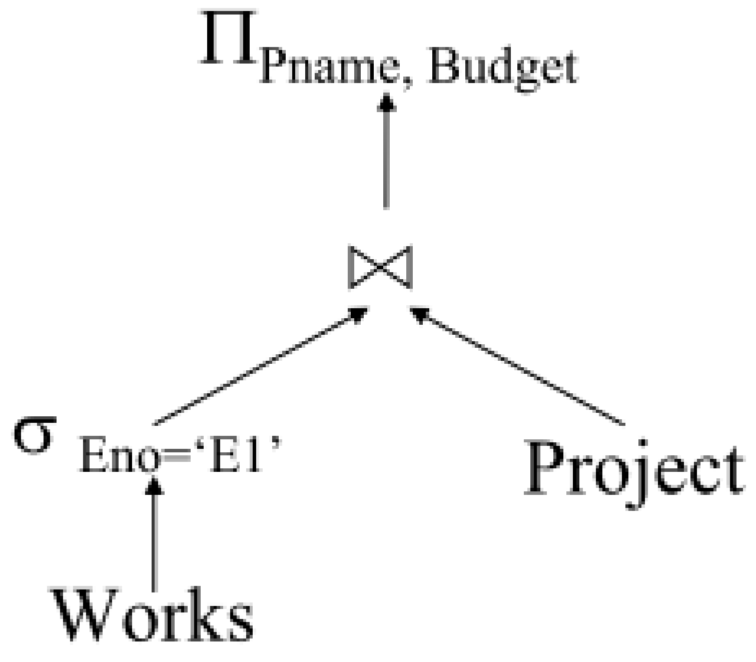
$$\Pi_{Pname,\ Budget}\ (Project \bowtie (\sigma_{Eno='E1'}(Works)))$$

$$\Pi_{Pname,\ Budget}(\sigma_{Project.Pno=Works.Pno}(Project \times {}_{\sigma\ Eno='E1}'\ (Works)))$$

Arbre algébrique

- Utile pour manipuler les requêtes (**optimisation**, **vues**)
- Exemple

$$\Pi_{Pname,\ Budget}(Project \bowtie (\sigma_{Eno='E1'}(Works)))$$



Mise à jour d'une BD relationnelle

- Avec les opérateurs relationnels
  - ***Insertion*** de n-uplets = ***union de la relation contenant*** les *n-uplets à insérer* avec ***la relation déjà existante*** dans la base
  - ***Suppression*** de n-uplets = ***différence entre la relation existante*** dans la base et ***la relation contenant les n-uplets à supprimer***
  - ***Modification*** = ***suppression*** suivie d'une **insertion**