Bases de Données

Fabrice.Jouanot (CM, TD, TP)
Nicolas Marihno (CM, TD, TP)
Catherine.Berrut (TD, TP)
kadir-uraz.alacali (TD, TP)
@univ-grenoble-alpes.fr)

- Objectifs
- Planning
- Plan du cours
- Evaluation

Objectifs

- Maîtriser le modèle relationnel de données et les langages associés
  - ✓ Algèbre relationnel
  - ✓ Langage SQL LMD & LDD
  - ✓ SQL, pattern de quêtes complexes
  - ✓ Normalisation d'un schéma
- Mise en œuvre d'applications de bases de données
  - ✓ Accès basiques (JDBC/DAO)
  - ✓ Mapping relationnel Object JPA (ORM)
  - ✓ Approche Serveur RestFull par micro-services: SpringBoot

Evaluation

- Contrôle continu:
  - Evaluation TPs (binômes)
  - 1 Devoir surveillé sur table et sur machine (note individuelle)
- Examen
- Support de cours sur Moodle (code BDSI26)

BD relationnelles

- Définition du modèle
- Contraintes liées au modèle
- Exemple de "schéma relationnel"

Base de données

- Définition
  - Ensemble cohérent, intégré, partagé de données structurées,
  - Définie pour les besoins d'une application.
- Système de gestion de base de données
  - Définir une représentation adaptée des informations
  - Garder les relations entre les données liées (data, texte ou média)
  - Stocker, interroger et manipuler (insérer, supprimer, mettre à jour)
  - Scalable: Gérer de très grandes quantités de données
  - Propriété transactionnel (ACID): Garantir la pérennité des données, accès concurrent, cohérence des données
- Objectif: Conserver la cohérence des données!

Historique du modèle

- C'est un modèle logique de niveau logique, très simple (peu de concepts et représentation tabulaire simple): repose sur une formalisation solide (ensemble + algèbre)
- Défini par Ted Codd en 1970; prix Turing en 1986. Développé par IBM lab.
- Aujourd'hui utilisé par tous les SGBD (Oracle, MySQL, PostgreSQL Sqlite …), **c'est le modèle de référence** !
- Le modèle repose sur deux concepts qui permettent une représentation tabulaire:
  - relation (≈ table), c'est le concept de base.
  - attribut typée (≈ colonne), une relation est composée d'attributs.
  - 12 règles de Codd

Concepts de base du modèle

Etudiant (N°Etud, nom, prénom, age)

nom de la relation

noms des attributs

schéma

population

tuple ou Occurrence (les tuples ne sont pas ordonnés)

Etudiant			
N°Etud	**nom**	**prénom**	**dateN**
136	Dupont	Jean	19/08/01
253	Aubry	Annie	20/12/00
101	Duval	André	21/06/02
147	Dupont	Marc	21/11/00

Définitions

- Un **domaine** est un ensemble de valeurs atomiques.

D1={Aubry, Dupont, Duval}
D2={André, Annie, Jean, Marc}
D3={date > 31/12/1999}
D4={entier≠0}

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

D4×D1×D2×D3={(1,Aubry,André,19/08/01); …}
La relation **Etudiant** est un sous-ensemble de ce produit !

- Un **attribut** indique le rôle joué par un domaine dans une relation.

On définit par exemples 4 attributs N°Etud, Nom, Prenom et Age.

Sans oublier leurs domaines:
dom(N°Etud)=D4; dom(Nom)=D1; dom(Prenom)=D2; dom(dateN)=D3

Définitions (suite)

- Un **schéma** de relation est défini par un nom et l'ensemble des attributs de la relation.

Exemples: Etudiant(N°Etud, Nom, Prenom, DateN)

On ajoute une description formelle pour définir la sémantique des attributs.

{(ne,n,p,d)$\in$Etudiant $\Leftrightarrow$ l'étudiant dont le numéro d'identification national est ne, ayant pour nom n et prénom p. Sa date de naissance est d.}

- Un prédicat est la(les) condition(s) que doit vérifier un n-uplet.

  Le prédicat apparaît en complément de la définition formelle. Il traduit une ou des contraintes d'intégrité.

{(ne,n,p,d)$\in$Etudiant $\Leftrightarrow$ l'étudiant dont… Cet étudiant appartient à l'UFR IM2AG. Le prénom correspond à son premier prénom.}

Schéma relationnel

- Une BD est composé d'un ensemble de relations:

  Le schéma d'une BD relationnelle est composé d'un ensemble de schémas de relation: R1, R2, …, Rx

- Chaque relation doit être formellement défini afin d'utiliser sans ambiguïté la notation simplifiée:

  Ri (A1, A2, …, Ay) avec Ai un attribut.

- Il faut suivre quelques règles pour obtenir un schéma correct…

Règles de structuration d'une relation

- Attributs: simples et monovalués (valeur atomique)
  - Respect de la première forme normale
- Une relation dispose d'une structure plate régulière (≠ objets)

tuple

x	y	z	w

x: une et une seule valeur atomique par attribut

x	y	z	w
		z	
		z	
xi	yi	zi	wi
			wi
			wi

INTERDIT

Règles d'identification

- Toute relation possède un identifiant (≡clé ou clé primaire)
  - L'identifiant est constitué d'attributs de la relation,
  - Il ne peut y avoir deux tuples identiques dans la même relation, c'est à dire possédant les mêmes valeurs pour les attributs constituant la clé.
- L'identifiant n'admet pas de valeurs NULL (!)

Etudiant2	**N°Etud**	**nom**	**prénom**	**dateN**
	136	Dupont	Jean	19/08/01
	253	Aubry	Annie	20/12/00
	101	Duval	André	21/06/02
	147	Dupont	Marc	21/11/00

Identifiant d'une relation

- Une relation peut avoir plusieurs identifiants, on parle d'identifiants primaires et secondaires (clés candidates).

  Etudiant2 (N°Etud, Nom, Prénom, dateN)
  - N°Etud est l'idientifiant.
  - Nom+Prénom peut être considéré comme un autre identifiant s'il n'y a pas d'homonyme dans la base.

  Dans ce cas il faut choisir l'un des deux comme identifiant (primaire).

- Définition: L'identifiant d'une relation est un ensemble **minimum** d'attributs de la relation, tel qu'il n'existe pas 2 tuples ayant même valeur pour cet identifiant.

Etudiant2 (N°Etud, Nom, Prénom, dateN)

La somme de tous les attributs n'est pas un ensemble minimum !

- Règle: tous les attributs de tout identifiant doivent toujours avoir une valeur connue (non nulle).

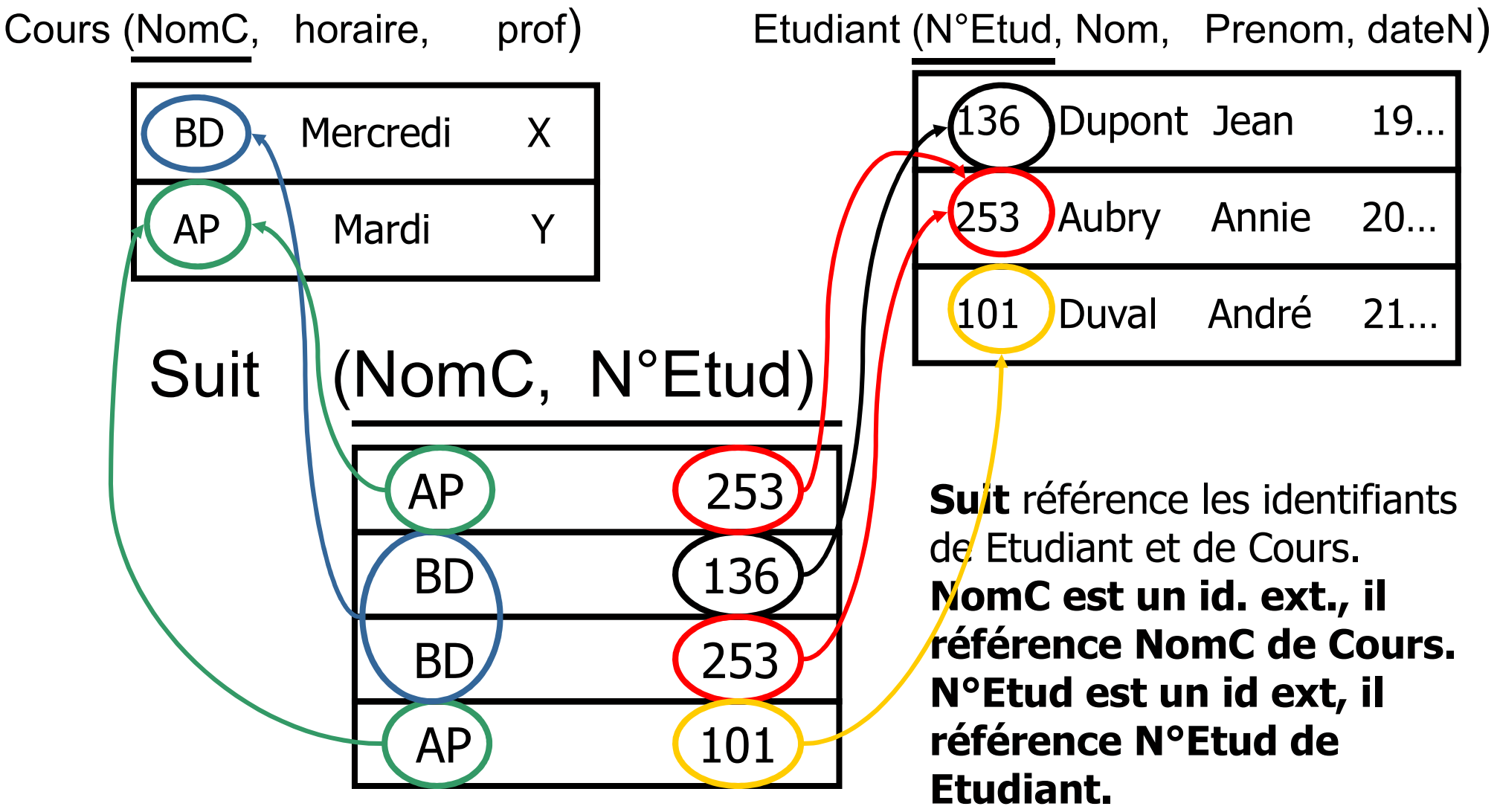
Identifiant externe où comment lier les informations (relations)

- On veut stoker le fait qu'un étudiant suit certains cours:
  - On peut tout mettre dans une table ⇒ Redondances (perte d'espace, gestion difficile, risque d'incohérence)

Etudiant_V2	N°Etud	nom	prénom	dateN	Cours	Prof	Horaire
	136	Dupont	Jean	19…	BD	X	mercredi
	253	Aubry	Annie	20…	AP	Y	jeudi
	101	Duval	André	21…	AP	Y	jeudi
	253	Aubry	Annie	20…	BD	X	mercredi

Attention il faut modifier l'identifiant

Identifiant externe par l'exemple



Définition d'une relation

- Une relation est donc définie par :
  - son nom
  - ses d'attributs
  - son (ses) identifiant(s) (souligné)
  - son (ses) identifiant(s) possible(s)
  - une définition formelle pour lever toutes ambiguïtés d'interprétation.
  - son (ses) identifiant(s) externe(s) (à insérer dans la définition formelle).

- Exemple :

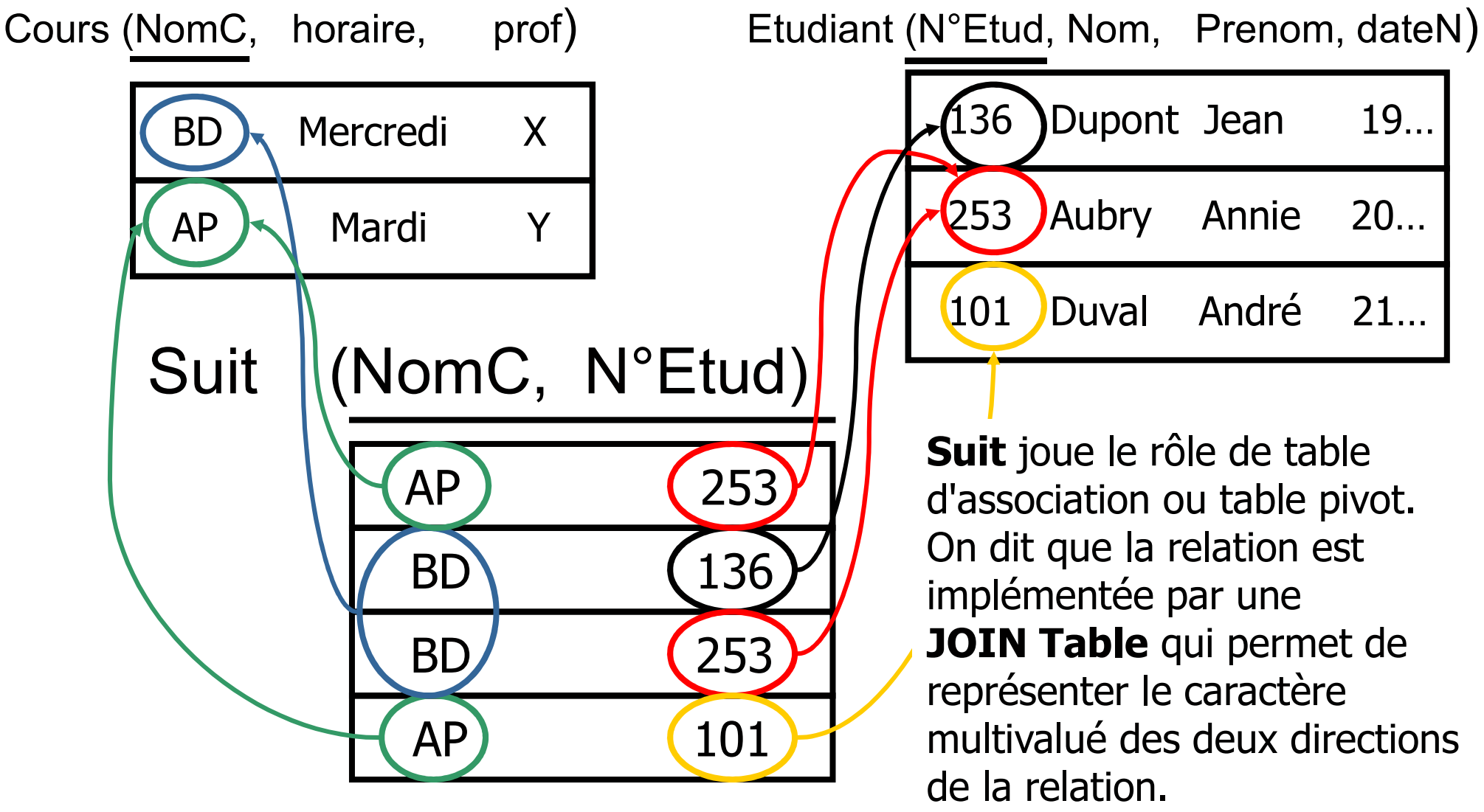
  Suit (<u>NomC, N°Etud</u>)

  {(n, ne)$\in$Suit $\Leftrightarrow$ l'étudiant identifié par le numéro ne suit le cours identifié par son nom n}.

  $\pi_{NomC}(Suit) \subset \pi_{NomC}(Cours)$

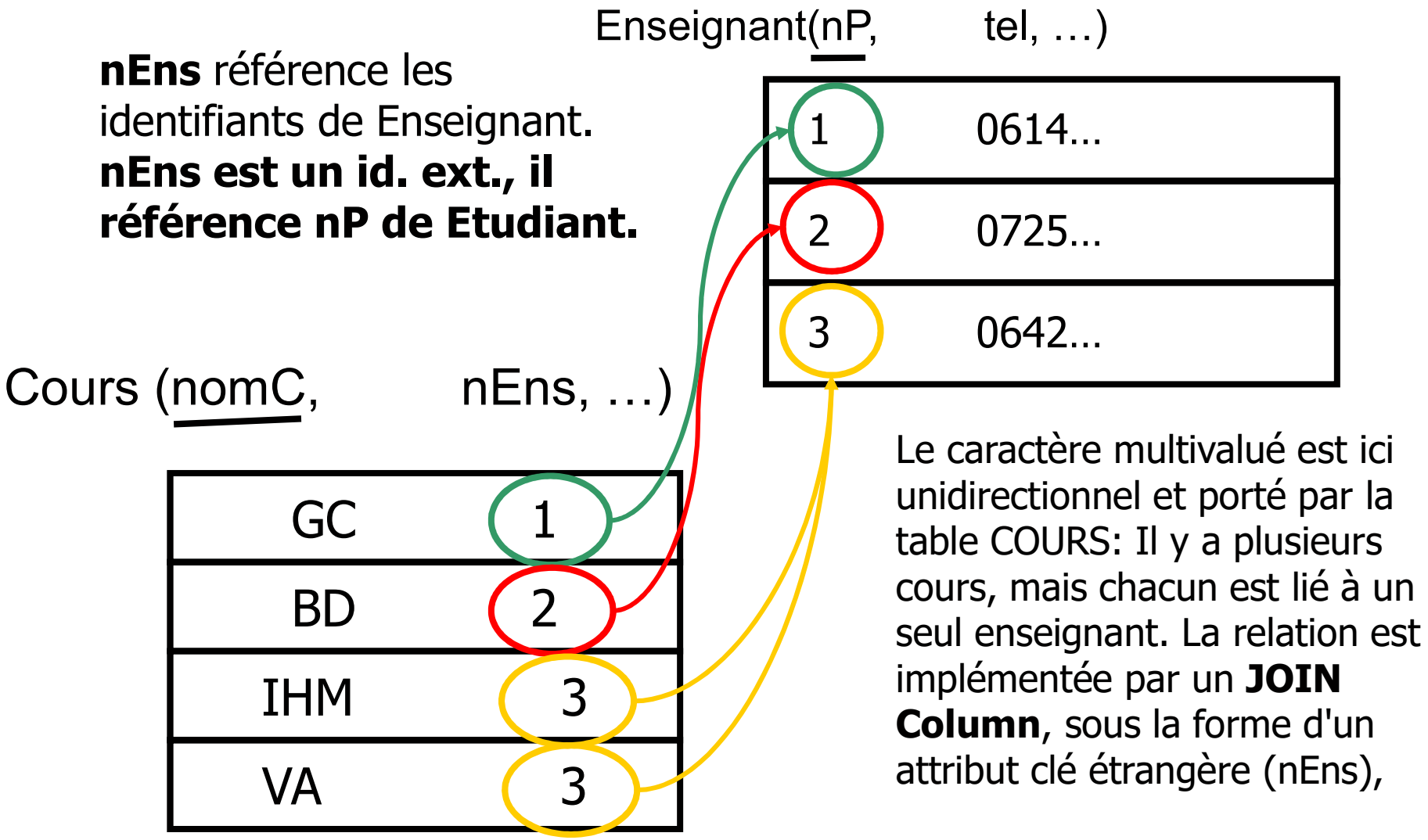
  $\pi_{N°Etud}(Suit) \subset \pi_{N°Etud}(Etudiant)$

Relation ManyToMany



Relation ManyToOne

nEns référence les identifiants de Enseignant. **nEns est un id. ext., il référence nP de Etudiant.**

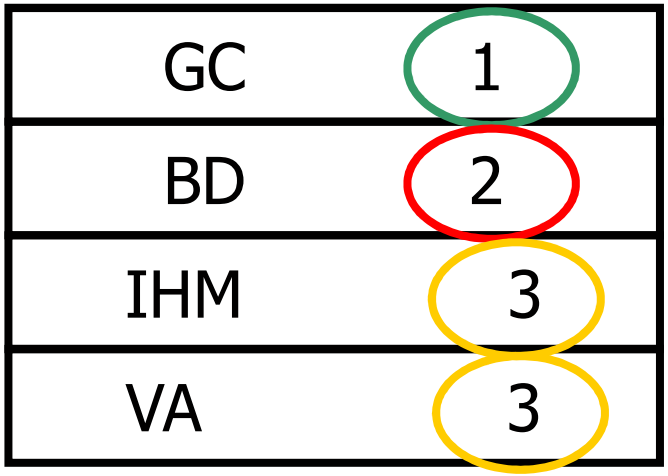


Le caractère multivalué est ici unidirectionnel et porté par la table COURS: Il y a plusieurs cours, mais chacun est lié à un seul enseignant. La relation est implémentée par un **JOIN Column**, sous la forme d'un attribut clé étrangère (nEns),

```
Select E.* from Cours C join Enseignant E on nEns=nP
Where nomC = 'BD'
```

Relation OneToMany

Cours (nomC, nEns, …)



Le lien inverse de la relation ManyToOne n'est pas implémenté explicitement. Pour accéder au caractère multivalué de la relation (d'un Enseignant on veut obtenir les Cours qui lui sont associés) il faut une requête sur cours, mais sans jointure.

```
Select C.* from Cours C where nEns = 3
```

Cette implémentation de la relation entre deux relations est particulièrement efficace dans le sens OneToMany !

Contraintes de modélisation:
Représentation du caractère multivalué

Exemple: mémoriser les différents prénoms des étudiants

- INCORRECTE (mais possible avec des limitations):
  Plusieurs attributs : Prénom1, Prénom2,…
- CORRECTE: créer une relation supplémentaire:
  EtudPrenoms (N°Etud, Prénom)
  {(n,p)∈EtudPrénoms ⇔ l'étudiant identifié par le numéro n possède le prénom p}.

136	Jean
136	Paul
101	André
253	Annie
253	Claudine

Ou avec liste ordonnée:

EtudPrenoms2 (N°Etud, N°Prénom, Prénom)
{(n,np,p)∈EtudPrénoms2 ⇔ l'étudiant identifié par le numéro n possède comme np^{ième} prénom le prénom p}.

Join Table ou Join Column ?

EtudPrenoms référence les identifiants de Etudiant.
N°EtudNomC est un id. ext., il référence N°etud de Etudiant.

Etudiant (N°Etud, Nom, dateN)

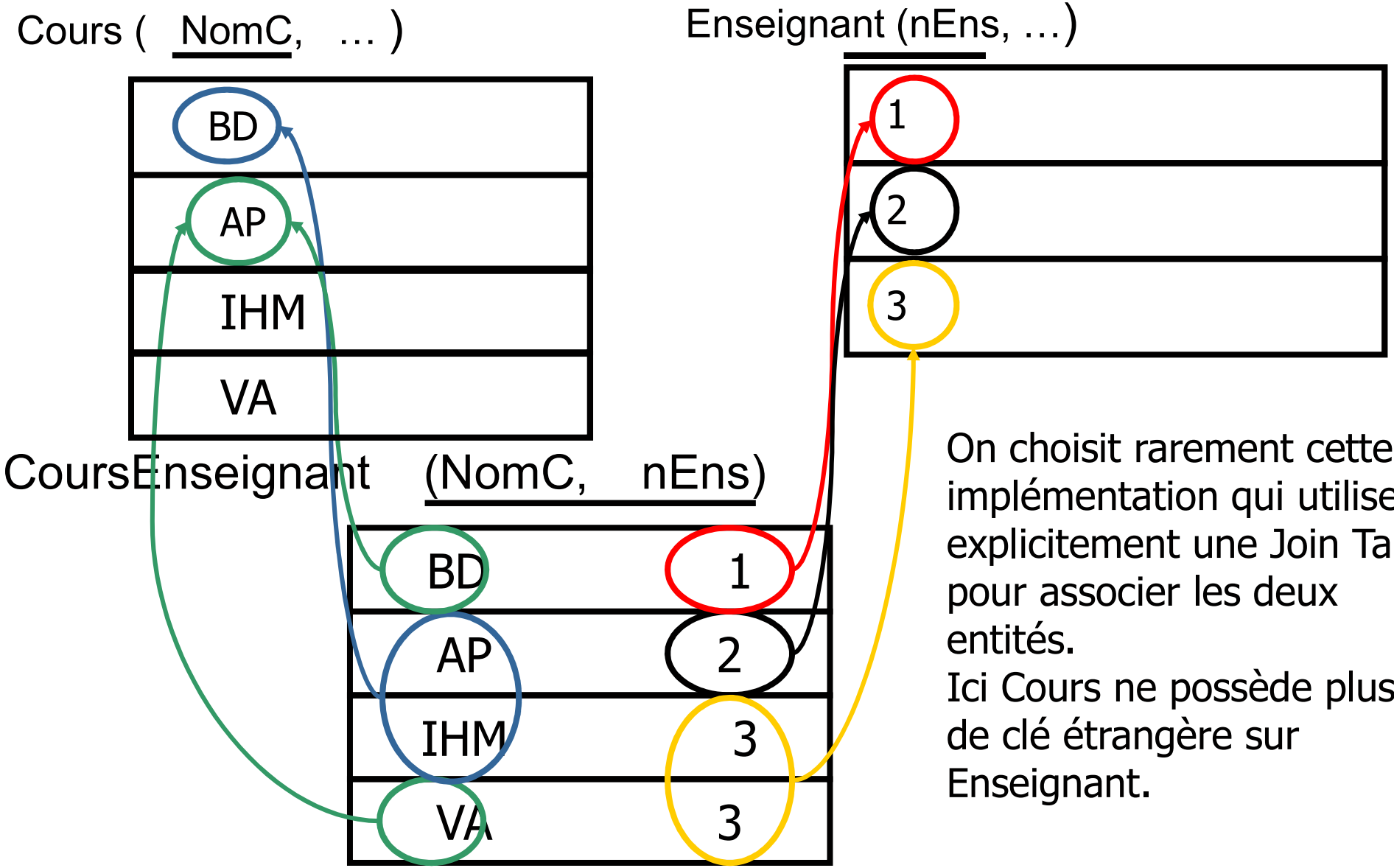
136	Dupont	19…
253	Aubry	20…
101	Duval	21…

EtudPrenoms (N°Etud, Prénom)

136	Jean
136	Paul
253	Annie
253	Claudine
101	André

Le caractère multivalué est bidirectionnel: Un étudiant possède plusieurs prénom, un prénom est possédé par plusieurs étudiants.
Nous sommes plus poche d'un Join Table que d'un Join Column: cas d'une collection de valeurs

Relation OneToMany / ManyToOne implémentée avec une Join Table



On choisit rarement cette implémentation qui utilise explicitement une Join Table pour associer les deux entités.
Ici Cours ne possède plus de clé étrangère sur Enseignant.

Contraintes de modélisation:
Représentation d'attribut complexe

Soit *Adresse : nomrue , num , ville, CP*

- Solution 1:
  - un attribut par composant: nomrue , num , ville, CP

  < "Rue Casimir Brenier", "2", "Grenoble", "38000" >
  - il est éventuellement possible de définir par ailleurs une vue restituant la notion globale d'adresse (cf cours SQL).
  - Pour indiquer un lien sémantique entre ces données on peut donner des infos dans le nom des attributs:
    Adr_nomrue , Adr_num , Adr_ville, Adr_CP
- Solution 2:
  - un attribut Adresse, domaine: chaine de caractères

  < "2, Rue de Casimir Brenier, 38000 Grenoble " >
  - La recherche sur un composant de l'adresse est plus difficile.

L’algèbre relationnelle

- Définition de l'algèbre relationnelle
- Les opérateurs de l'algèbre
- Exemple de requêtes algébriques

Algèbre relationnelle

- Une algèbre est un ensemble d'opérateurs de base, formellement définis, qui peuvent être combinés à souhait pour construire des calculs sur les données.
- Fermeture. Une algèbre est dite fermée si le résultat de tout opérateur est du même type que les opérandes (ce qui est indispensable pour construire des expressions relationnelles).
- Complétude. Toute manipulation pouvant être souhaitée par les utilisateurs devrait pouvoir être exprimable par une expression algébrique.

L'algèbre relationnelle en 5 points

- **Opérandes**: relations du modèle relationnel
- **Fermeture**: le résultat de toute opération est une nouvelle relation
- **Complétude**: permet toute opération sauf les fermetures transitives (cf. Normalisation)
- **Opérations unaires** (une seule opérande):
  sélection (noté σ), projection (π), renommage (ρ)
- **Opérations binaires**:
  produit cartésien ($\times$), jointures ($\bowtie \ltimes \rtimes \overline{\bowtie} \triangleright$), union ($\cup$), intersection ($\cap$), différence ($-$), division ($\div$)

- Pour chacune de ces 9 opérations, vous trouverez: un exemple, la représentation formelle, la représentation graphique.

Sélection

- But: ne retenir que certains tuples dans une relation (filtrage)

Personne	**nP**	**Nom**	**Adr**
	1	Dupont	Grenoble
	2	Aubry	Dijon
	3	Duval	Grenoble

On ne veut que les personnes dont la valeur de Adr est Grenoble:

Grenoblois = $\sigma_{\text{Adr = 'Grenoble'}}$**(Personne)**

Grenoblois	**nP**	**Nom**	**Adr**
	1	Dupont	Grenoble
	3	Duval	Grenoble

Sélection

- Syntaxe: $\sigma_{p}(R)$(*opération unaire*)
  p: prédicat de sélection (condition de sélection)

  p=< prédicat-élémentaire opérateur-logique prédicat-élémentaire >
  - opérateur-logique $\in \{ \wedge, \vee \}$
  - prédicat-élémentaire :

  <[$\neg$] attribut opérateur-de-comparaison constante|attribut>
    - attribut est un attribut de la relation R
    - opérateur-de-comparaison $\in \{=, \neq, <, >, \leq, \geq\}$

- sémantique : crée une nouvelle relation dont la population est l'ensemble des tuples de R qui satisfont le prédicat p.
- schéma ($\sigma_{p}(R)$) = schéma (R)
- population ($\sigma_{p}(R)$) $\subseteq$ population (R)

Projection

- But: ne retenir que certains attributs dans une relation

Personne	(**nP**	**Nom**	**Adr)**
	1	Dupont	Grenoble
	2	Aubry	Dijon
	3	Duval	Grenoble

On ne veut que les attributs Nom et Adr:

Identité = $\pi_{\text{Nom, Adr}}$(Personne)

Identité	(**Nom**	**Adr)**
	Dupont	Grenoble
	Aubry	Dijon
	Duval	Grenoble

Projection

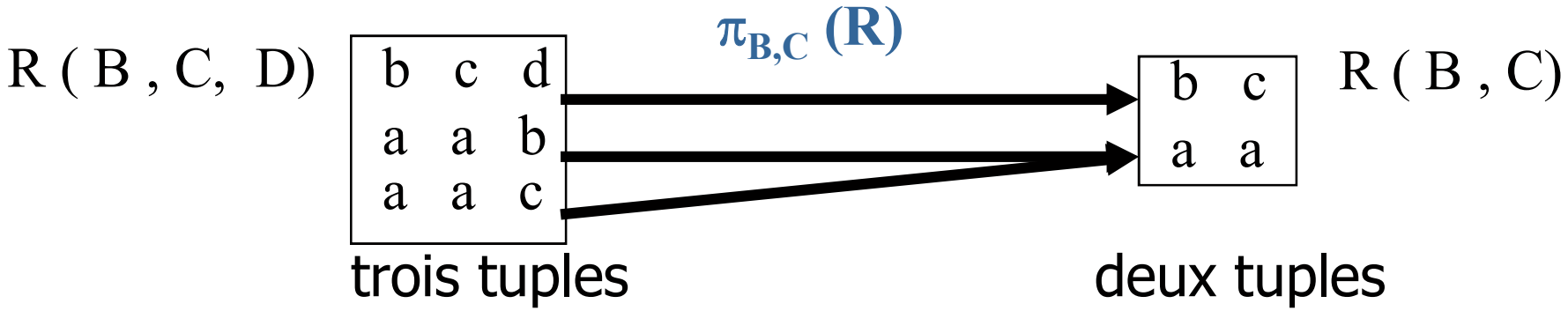
- syntaxe: $\pi_{\mathbf{Attrs}}$ **(R)** (*opération unaire*)

  Attrs: ensemble des attributs de R à conserver dans le résultat

- sémantique : crée une nouvelle relation dont la population est l'ensemble des tuples de R et dont le schéma est réduit aux attributs explicités.

- schéma ($\pi_{\mathbf{Attrs}}$ **(R)**) $\subseteq$ schéma (R)

- nb tuples ($\pi_{\mathbf{Attrs}}$ **(R)**) $\leq$ nb tuples (R) : Elimination des doublons
  - une projection qui ne conserve pas la clé de la relation peut générer dans le résultat deux tuples identiques mais le résultat ne gardera que des tuples différents (une relation est un ensemble).



Combinaison des opérateurs: Sélection-projection

- On veut l'identité des grenoblois:
  - **Grenoblois = $\sigma_{Adr = 'Grenoble'}$(Personne)**
  - **Id_Grenoblois = $\pi_{Nom, Adr}$(Grenoblois)**

OU Id_Grenoblois = $\pi_{Nom, Adr}(\sigma_{Adr = 'Grenoble'}$(Personne))

Id_Grenoblois	(**Nom**	**Adr)**
	Dupont	Grenoble
	Duval	Grenoble

Renommage

- but: résoudre des problèmes de compatibilité entre noms d'attributs de deux relations, opérandes d'une opération binaire
- opération unaire
- syntaxe :
  - $\rho_{\text{nouveau\_nom\_de\_Relation}}$(ancien_nom_de_Relation)
    - Hors d'une expression, on peut aussi construire une nouvelle relation en résultat d'une expression: S = R
  - $\rho_{\text{nouveau\_nom\_d'attribut} \leftarrow \text{ancien\_nom\_d'attribut}}$(Table)
- schéma : ne modifie pas la structure du schéma.
- précondition : le nouveau nom d'attribut n'existe pas déjà.
- exemple : R2 = $\boldsymbol{\rho_{C \leftarrow B}(R1)}$

R1

A	B
a	b
y	z
b	b

R2

A	**C**
a	b
y	z
b	b

Produit cartésien

- but: construire toutes les combinaisons possibles de tuples de deux relations.
- syntaxe : R × S (opération binaire)
- Sémantique : chaque tuple de R est combiné avec tous les tuples de S. *Si R et S ont des attributs de même nom alors on doit renommer avant l'opération, mais par convention on considère souvent que les attributs sont préfixés par le nom de la table initiale pour éviter les renommages.*
- schéma : schéma (R × S) = schéma(R) ∪ schéma(S)
- Exemple :

R

A	B
a	b
b	c
c	b

n tuples

S

C	D	E
c	d	e
b	a	b
a	a	c

m tuples

R × S

A	B	C	D	E
a	b	c	d	e
a	b	b	a	b
a	b	a	a	c
b	c	c	d	e
b	c	b	a	b
b	c	a	a	c
c	b	c	d	e
c	b	b	a	b
c	b	a	a	c

n x m tuples

Produit cartésien: exemple "bête"

- Faire le produit cartésien "bête" de deux relations

Personne

(**nP**	**Nom**	**Adr)**
1	Dupont	Grenoble
2	Aubry	Dijon
3	Duval	Grenoble

PersonnePrénoms

(**num**	**Prénom** **)**
1	Jean
1	Chantal
2	André
3	René

bête = Personne × PersonnePrénoms

(nP	**Nom**	**Adr**	**num**	**Prénom)**
1	Dupont	Grenoble	1	Jean
1	Dupont	Grenoble	1	Chantal
1	Dupont	Grenoble	2	André
1	Dupont	Grenoble	3	René
2	Aubry	Dijon	1	Jean
2	Aubry	Dijon	1	Chantal
2	Aubry	Dijon	2	André
2	Aubry	Dijon	3	René
3	Duval	Grenoble	1	Jean

etc…

Produit cartésien: exemple "intelligent"

- Faire le produit cartésien "intelligent" des relations Personne et PersonnePrénoms en reliant une personne à ses prénoms.

Personne

(**nP**	**Nom**	**Adr)**
1	Dupont	Grenoble
2	Aubry	Dijon
3	Duval	Grenoble

PersonnePrénoms

(num	**Prénom)**
1	Jean
1	Chantal
2	André
3	René

intelligent = $\sigma_{nP=num}$**(Personne** $\times$ **PersonnePrénoms)**

(nP	**Nom**	**Adr**	**num**	**Prénom)**
1	Dupont	Grenoble	1	Jean
1	Dupont	Grenoble	1	Chantal
2	Aubry	Dijon	2	André
3	Duval	Grenoble	3	René

- On a reconnu le résultat d'une jointure !

Jointure: une grande famille

- but: Réaliser une combinaison sémantique des tuples de deux tables. La jointure des tuples de deux tables doit vérifier un critère de jointure (contraintes sur les valeurs d'attributs des deux tables).

Personne

(**nP**	**Nom**	**Adr)**
1	Dupont	Grenoble
2	Aubry	Dijon
3	Duval	Grenoble

PersonnePrénoms

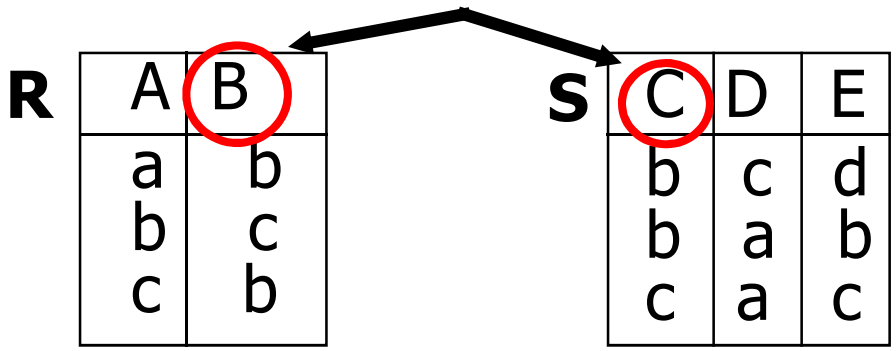
(**num**	**Prénom)**
1	Jean
1	Chantal
2	André
3	René

Personne $\bowtie_{nP=num}$ **PersonnePrénoms**

(nP	**Nom**	**Adr**	**num**	**Prénom)**
1	Dupont	Grenoble	1	Jean
1	Dupont	Grenoble	1	Chantal
2	Aubry	Dijon	2	André
3	Duval	Grenoble	3	René

(Theta)Jointure

- syntaxe : $R \bowtie_p S$
  - R et S sont deux relations a priori relié sémantiquement par p.
  - p: prédicat ou critère de jointure, c'est une expression booléenne composée de prédicats élémentaires combinés par une opérateur logique:
    p=< prédicat-élémentaire $\wedge \mid \vee$ prédicat-élémentaire >
    prédicat-élémentaire= < attribut op_comparaison attribut>
    op_comparaison $\in \{=, \neq, <, >, \leq, \geq\}$
  - $R \bowtie_p S = \{<r,s> / r \in R \wedge s \in S \wedge p\}$ avec Domine(r)$\subseteq$Domine(s) (même signification)
  - schéma $(R \bowtie_p S)$ = schéma (R) $\cup$ schéma (S)
  - **C'est une opération binaire équivalente à un produit cartésien suivi d'une sélection:** $R \bowtie_p S \approx \sigma_p(R \times S)$



R $\bowtie_{B \neq C}$ **S**

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

Jointure naturelle, la plus fréquente

- but: R ⋈ S combine les tuples de R et S en considérant l'égalité des attributs en commun (prédicat de jointure implicite).

Personne

(**nP**	**Nom**	**Adr)**
1	Dupont	Grenoble
2	Aubry	Dijon
3	Duval	Grenoble

PP2= $\rho_{\mathbf{nP} \leftarrow \text{num}}$ **(PersonnePrénoms)**

(nP	**Prénom)**
1	Jean
1	Chantal
2	André
3	René

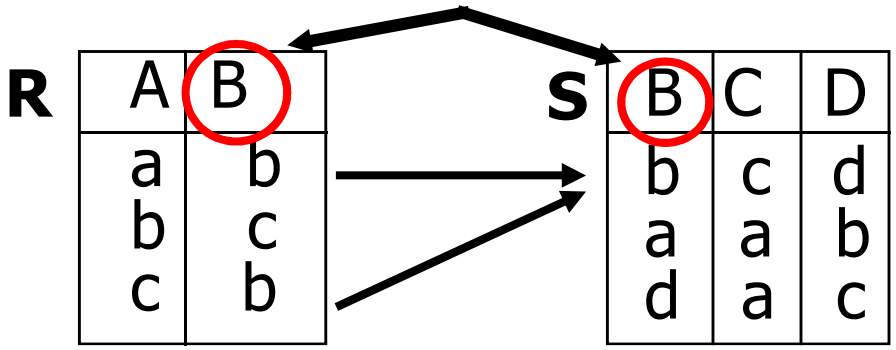
Personne ⋈ PP2

(nP	**Nom**	**Adr**	**Prénom)**
1	Dupont	Grenoble	Jean
1	Dupont	Grenoble	Chantal
2	Aubry	Dijon	André
3	Duval	Grenoble	René

Jointure naturelle

- syntaxe : $R \bowtie S$
  - Soit $R(X_1 \dots X_n)$ et $S(X_{n-p} \dots X_m)$
    $R \bowtie S = \{ <a,b,c> / r \in R \wedge s \in S \wedge a = \pi_{X1..\, Xn-p-1}(R)$
    $\wedge\ b = \pi_{Xn-p \dots\, Xn}(R)$
    $\wedge\ c = \pi_{Xn+1 \dots\, Xm}(S)$
    $\wedge\ \pi_{Xn-p \dots\, Xn}(R) = \pi_{Xn-p \dots\, Xn}(S) \}$
  - L'opération est équivalente à une θjointure dont le prédicat définit une égalité de valeurs entre les attributs de même noms entre R et S:
    $R \bowtie S = (R \bowtie_{\mathbf{R.Xn-p=S.Xn-p \dots \wedge R.Xn=S.Xn}} S)[R.X_1 \dots R.X_n, S.X_{n+1} \dots S.X_m]$

- schéma : schéma $(R \bowtie S)$ = schéma $(R) \cup$ schéma (S)
  - les attributs de même nom n'apparaissent qu'une seule fois !
  - si $\neg p$ (prédicat non vérifié): pas d'attributs en commun alors $R \bowtie S = \varnothing$



R ⋈ S

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

Semi-Jointure, Quand une des tables sert à filtrer

- but: R ⋉ S filtre les tuples d'une table R par jointure avec S. Seule les tuples de R sont conservées. *Très utile en optimisation et dans les systèmes distribués.*

Etudiant

(**nEtud**	**Nom**	**Prenom**	**DateN)**
136	Dupont	Jean	2005
253	Aubry	Annie	2002
101	Duval	André	2004

Suit

(NomC	**nEtud)**
AP	253
BD	136
BD	253
AP	101

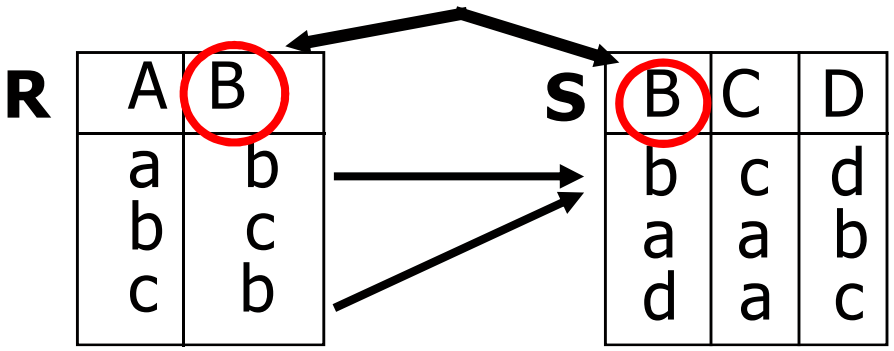
Etudiant ⋉ $\sigma_{\text{NomC="BD"}}$(Suit)

(nEtud	**Nom**	**Prenom**	**DateN)**
136	Dupont	Jean	2005
253	Aubry	Annie	2002

Semi-Jointure

- syntaxe : $R \ltimes S$ ou $S \rtimes R$
  - Soit $R(X_1 \ldots X_n)$ et $S(X_{n-p} \ldots X_m)$
    $R \ltimes S = \{ <a,b> / r \in R \wedge s \in S \wedge a = \pi_{X1..\, Xn-p-1}(R)$
    $\wedge\ b = \pi_{Xn-p \ldots\, Xn}(R)$
    $\wedge\ \pi_{Xn-p \ldots\, Xn}(R) = \pi_{Xn-p \ldots\, Xn}(S)\}$
  - L'opération est équivalente à une jointure naturelle avec projection sur les attributs de R

- schéma : schéma $(R \ltimes S)$ = schéma$(S \rtimes R)$ = schéma (R)
  - les attributs de même nom n'apparaissent qu'une seule fois !
  - si $\neg p$ (prédicat de jointure non vérifié): $R \ltimes S = \varnothing$



R $\ltimes$ **S**

A	B
a	b
c	b

Anti-Jointure, quand l'absence de jointures m'intéresse (idée de AUCUN)

- but: une anti-jointure de R ▷ S calcule les lignes d'une table de R qui n'ont pas de correspondance dans la table S*. Le résultat contient exclusivement des valeurs non jointes*

Etudiant

(**nEtud**	**Nom**	**Prenom**	**DateN)**
136	Dupont	Jean	2005
253	Aubry	Annie	2002
101	Duval	André	2004

Suit

(NomC	**nEtud)**
AP	253
BD	136
BD	253

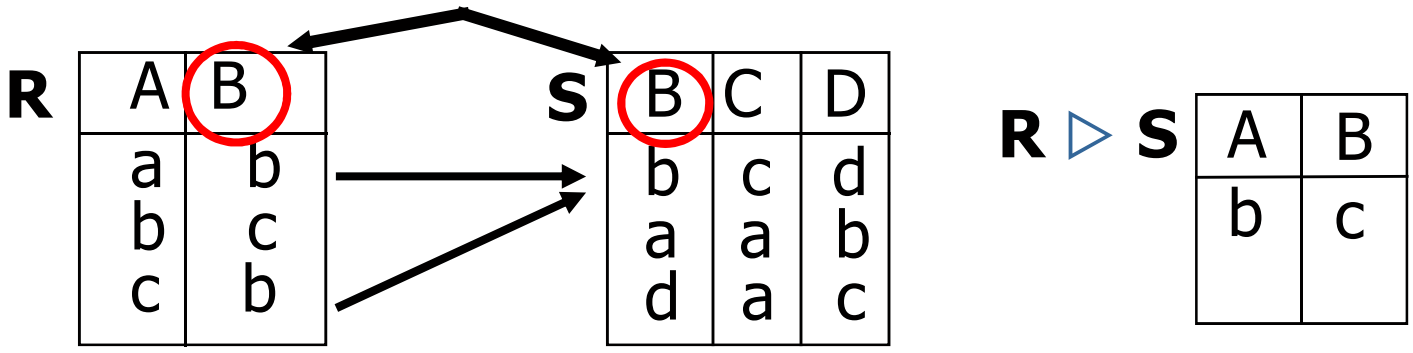
Etudiant ▷ Suit

(nEtud	**Nom**	**Prenom**	**DateN)**
101	Duval	André	2004

Anti-Jointure

- syntaxe : $R \triangleright S$
  - Soit $R(X_1 \ldots X_n)$ et $S(X_{n-p} \ldots X_m)$
    $R \triangleright S = R - \{ <a,b> / r \in R \wedge s \in S \wedge a = \pi_{X1..\, Xn-p-1}(R)$
    $\wedge\ b = \pi_{Xn-p \ldots\, Xn}(R)$
    $\wedge\ \pi_{Xn-p \ldots\, Xn}(R) = \pi_{Xn-p \ldots\, Xn}(S)\}$
  - L'opération est équivalente à une différence entre R et $R \ltimes S$

- schéma : schéma $(R \triangleright S)$ = schéma (R)
  - les attributs de même nom n'apparaissent qu'une seule fois !
  - si p (prédicat de jointure) vérifié $\forall r \in R$: $R \triangleright S = \varnothing$



Jointure Externe, Obtenir un résultat même en absence de jointure

- but: une jointure externe de R ⟕ S calcule les lignes d'une table de S qui se combine avec R selon un prédicat de jointure (implicite/naturelle ou explicite). *En l'absence de jointure pour un tuple de R, les champs provenant de S prennent la valeur NULL.*

Etudiant

(**nEtud**	**Nom**	**Prenom**	**DateN)**
136	Dupont	Jean	2005
253	Aubry	Annie	2002
101	Duval	André	2004

Suit

(NomC	**nEtud(**
AP	253
BD	136
BD	253

Etudiant ⟕ Suit

(nEtud	**Nom**	**Prenom**	**DateN**	**NomC)**
136	Dupont	Jean	2005	AP
253	Aubry	Annie	2002	AP
253	Aubry	Annie	2002	BD
101	Duval	André	2004	NULL

Jointure Externe

- syntaxe : $R \rtimes S$
  - Soit $R(X_1 \ldots X_n)$ et $S(X_{n-p} \ldots X_m)$
    $R \rtimes S = \{ <a,b> / r \in R \wedge s \in S \wedge a = \pi_{X1..\, Xn-p-1}(R)$
    $\wedge\ b = \pi_{Xn-p \ldots\, Xn}(R)$
    $\wedge\ \pi_{Xn-p \ldots\, Xn}(R) = \pi_{Xn-p \ldots\, Xn}(S)\}$
    $\cup$
    $(R \triangleright S) \times \{<c> / c = \pi_{Xn-p \ldots\, Xm}(S) \wedge\ <c>=<NULL>\}$
    }
  - L'opération est équivalente à une union entre $R \bowtie S$ et le produit cartésien de $R \triangleright S$ avec un tuple ayant le même schéma que S mais avec tous les attributs à NULL,

- schéma : schéma $(R \rtimes S)$ = schéma $(R \bowtie S)$
  - les attributs de même nom n'apparaissent qu'une seule fois !
  - si $\neg p$ (prédicat de jointure non vérifié): $card(R \rtimes S) = card(R)$

R

A	B
a	b
b	c
c	b

S

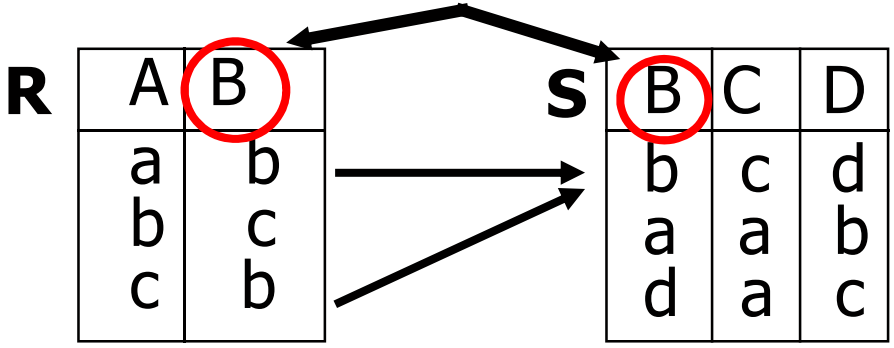
B	C	D
b	c	d
a	a	b
d	a	c

R $\rtimes$ **S**

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

Jointure Externe

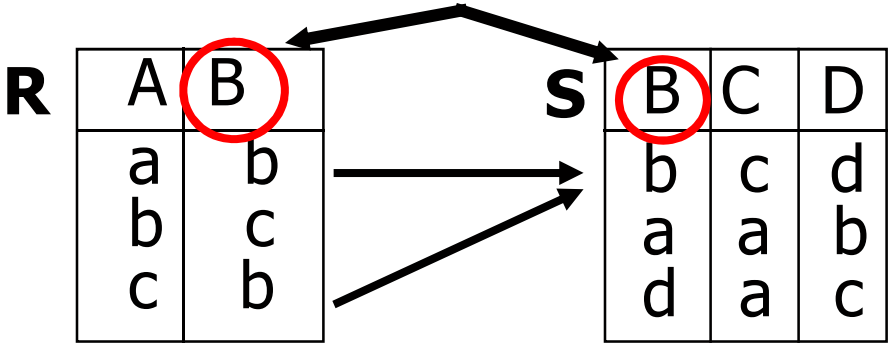
- La jointure externe est directionnelle: identifie la table à conserver
  - Externe gauche: ⟕ (déjà vu)
  - Externe droite: ⟖



R ⟖ S

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

  - Externe gauche+droite: ⟗



R ⟗ S

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

Union

- but: Combiner les tuples de deux relations dans une seule relation.

Personne1

nP	Nom	Adr
1	Dupont	Grenoble
2	Aubry	Dijon
3	Duval	Grenoble

Personne2

nP	Nom	Adr
10	Rut	Grenoble
11	JeanJean	Grenoble

union = Personne1 ∪ Personne2

nP	Nom	Adr
1	Dupont	Grenoble
2	Aubry	Dijon
3	Duval	Grenoble
10	Rut	Grenoble
11	JeanJean	Grenoble

- Aurions nous pu réalisé l'union suivante ?

union2 = Personne ∪ PersonnePrénoms

Union

- C'est une opération binaire, syntaxe : $R \cup S$
  - $R \cup S = \{t/\ t \in R \vee t \in S\}$
  - réunit dans une même relation les tuples de R et ceux de S en supprimant les doublons.
- schéma($R \cup S$) = schéma(R) = schéma(S)
- précondition : schéma(R) = schéma(S) (sinon renommage !)
- exemple général:

R1

A	B
a	b
b	b
y	z

R2

A	B
u	v
y	z

R1 ∪ R2

A	B
a	b
b	b
y	z
u	v

Intersection

- but : Retenir uniquement les tuples communs à deux relations.

Personne1

nP	Nom	Adr
1	Dupont	Grenoble
2	Aubry	Dijon
3	Duval	Grenoble

Personne2

nP	Nom	Adr
2	Aubry	Dijon
11	JeanJean	Grenoble
3	Duval	Grenoble

intersection = Personne1 ∩ Personne2

nP	Nom	Adr
2	Aubry	Dijon
3	Duval	Grenoble

- C'est donc une opération binaire, syntaxe: $R \cap S$
  - $R \cap S = \{t/\ t \in R \wedge t \in S\}$
  - sélectionne les tuples à la fois dans R et S,
- schéma $(R \cap S)$ = schéma (R) = schéma (S)
- précondition : schéma (R) = schéma (S) (sinon renommage !)

Différence

- but: enlever d'une relation les tuples d'une autre.

Personne1

nP	Nom	Adr
1	Dupont	Grenoble
2	Aubry	Dijon
3	Duval	Grenoble

Personne2

nP	Nom	Adr
2	Aubry	Dijon
11	JeanJean	Grenoble
3	Duval	Grenoble

différence = Personne1 – Personne2

nP	Nom	Adr
1	Dupont	Grenoble

- c'est opération binaire, syntaxe: R – S
  - $R–S=\{t/\ t \in R \wedge t \notin S\}$
  - retient les tuples de R en y autant les tuples de S.
- schéma (R – S) = schéma (R) = schéma (S)
- précondition : schéma (R) = schéma (S) (sinon renommage !)

La Division

- but: R ÷ S calcule le sous-ensemble de n-uplets de R complété par tous les n-uplets de S. Permet de répondre aux requêtes du genre donner «les étudiants qui sont inscrits à TOUS les cours du cycle 1». *On veut les tuples d'une relation qui sont associés à l'ensemble complet des tuples d'une autre relation (quantificateur universel)*

Inscrit

nE	**nomC**
1	M32a
1	M32b
2	M31
3	M32a
3	M32b
3	M31

CoursCycle1

nomC
M32a
M32b
M31

division = Inscrit ÷ CoursCycle1

nE
3

Remarque: les attributs du dividende faisant partie du diviseur disparaissent du résultat !

La division

- Opération binaire, Syntaxe: R ÷ S
  - soient $R(A_1, \ldots, A_n)$ et $S(A_1, \ldots, A_m)$
    avec n>m et $A_1, \ldots, A_m$ un ensemble d'attributs de même nom dans R et S
    $R/V = \{ <a_{m+1}, a_{m+2}, \ldots, a_n> / \forall <a_1, a_2, \ldots, a_m> \in S, <a_1, a_2, \ldots, a_m, a_{m+1}, a_{m+2}, \ldots, a_n> \in R\}$
- schéma (R ÷ S) = schéma (R) – schéma (S)
- précondition: schéma(S) $\subset$ schéma(R)
- exemples :

R

A	B	C
1	1	1
1	2	0
1	2	1
1	3	0
2	1	1
2	3	3
3	1	1
3	2	0
3	2	1

V

B	C
1	1
2	0

R ÷ V

A
1
3

V'

B	C
1	1

R ÷ V'

A
1
2
3

V''

B	C
3	5

R ÷ V''

A
∅

Exemples de requêtes algébriques

En considérant le schéma FormaPerm:

Personne(nP, nom, adr)

PersonnePrénoms(num, prénom)

Etudiant(nP, nE, dateN)

EtudiantEtudes(nE, année, diplôme)

Enseignant(nP, tel, statut, banque, agence, compte)

Cours(nomC, cycle, nEns)

Obtenu(nE, nomC, note, année)

Inscrit(nE, nomC)

Prérequis(nomC, nomCprérequis)

- *Quel est le nom des personnes qui sont à la fois étudiant et enseignant (thèsard?)*
- *Quel est le nom des enseignants qui donne des cours de BD*

Exemples de requêtes algébriques

- *Quel est le nom des personnes qui sont à la fois étudiant et enseignant (thèsard?):*

  R1a= π_{nP}(Etudiant)

  R1b= π_{nP}(Enseignant)

  R1c=R1a $\cap$ R1b

  R1d=R1c $\rtimes$ Personne

  R1e= π_{nom}(R1d)

  ou bien en combinant les opérateurs:

  R1= $\pi_{nom}((\pi_{nP}$ (Etudiant) $\cap$ π_{nP} (Enseignant)) $\rtimes$ Personne)

- *Quel est le nom des enseignants qui donne des cours de BD:*

  R2a= $\sigma_{nomC='BD'}$ (Cours)

  R2b=Personne $\bowtie_{nP=nEns}$ R2a

  R2c= π_{nom}(R2b)

  ou bien

  R2= π_{nom}(Personne $\bowtie_{nP=nEns}$ ($\sigma_{nomC='BD'}$ (Cours)))

Propriétés et limitations de l'algèbre

- $R \cap S = S \cap R$, $R \bowtie S = S \bowtie R$, etc.
- $\sigma_{p1}(\sigma_{p2}(R)) = \sigma_{p2}(\sigma_{p1}(R)) = \sigma_{p1 \wedge p2}(R)$
- $\sigma_p(\pi_a R) = \pi_a(\sigma_p(R))$ ssi attributs(p) $\subseteq$ a
- $R \cap S = R - (R - S) = S - (S - R)$

➢ Limitations:

- Absence de partition et d'agrégation: "Pour chaque étudiant, donner le nombre d'inscription à des cours", "Donner l'âge moyen des enseignants."
- Pas de notion de récursivité: "Donner tous les cours prérequis par le prérequis d'un cours!".