

# Modèle relationnel et algèbre relationnelle

---

# Plan du document

- Modèle relationnel
- Opérateurs de l'algèbre
- Exemples de requêtes

# Les concepts descriptifs

- Notion de domaine
- Produit cartésien
- Relation
- Attribut
- Clé
- Schéma de relation et de BD
- Clé étrangère
- Métabase

- Définition

- Ensemble de valeurs

- Exemples

- Entier, réel, chaîne de caractères, booléen
  - Salaire = 1000...100000 (€)
  - Couleur = {'rosé', 'blanc', 'rouge'}

# Produit cartésien

## ■ Définition

- Le produit cartésien de  $D_1, \dots, D_n$  est l'ensemble des n-uplets (tuples)  $\langle V_1, \dots, V_n \rangle$  tel que  $V_i \in D_i$

## ■ Notation

- $D_1 \times \dots \times D_n$

## ■ Exemple :

- $D_1 = \{\text{'BD'}, \text{'SI'}\}$  (Code UV)
- $D_2 = \{\text{'Univ-Bougie'}, \text{'Univ-jijel'}, \text{'Assil'}\}$  (Prof)

$D_1 \times D_2$	$D_1$	$D_2$
	BD	Univ-Bougie
	BD	Univ-jijel
	BD	Assil
	SI	Univ-Bougie
	SI	Univ-jijel
	SI	Assil

# Relation

## ■ Définition

- ❑ Sous-ensemble du produit cartésien d'une liste de domaines
- ❑ Caractérisée par un nom

## ■ Exemple

- ❑  $D_1 = \text{Code UV}$
- ❑  $D_2 = \text{Prof}$

UV	$D_1$	$D_2$
	BD	Univ-Bougie
	SI	Univ-jijel

## Relation (2)

- Plus simplement, une relation est un tableau à deux dimensions
- Une ligne est un n-uplet (tuple)
- On associe un nom à chaque colonne afin de la repérer indépendamment de l'ordre = attribut
  - Prend ses valeurs dans un domaine
  - Exemple : code

UV	code	coord
	BD	Univ-Bougie
	SI	Univ-jijel

# Exemples de relations

Elève	Num	Nom	Adresse	Age
	1	Bélaïd	Jijel	20
	2	Assil	Bougie	20
	3	Taleb	Jijel	21

Inscrit	NumElève	CodeUV	Note
	2	BD	10
	1	BD	20
	2	SI	17
	3	SI	18



# Clé

## ■ Définition

- ❑ Une clé est un **groupe d'attributs minimum** qui détermine un n-uplet unique dans une relation (à tout instant)

## ■ Exemple

- ❑ Clé de Elève ?
- ❑ Clé de UV ?
- ❑ Clé de Inscrit ?

## ■ Contrainte d'intégrité

- ❑ Toute relation doit posséder une clé renseignée (sans valeur inconnue)

# Schéma de relation

## ■ Définition

- ❑ Le schéma d'une relation décrit :
  - Son nom
  - La liste des attributs qu'elle comporte et des domaines associés
  - La liste des attributs composant la clé (la clé est soulignée)

## ■ Exemple

- ❑ Elève(num : entier, nom : chaîne, adresse : chaîne, age : entier de 18 à 35)

## ■ Intention vs. Extension

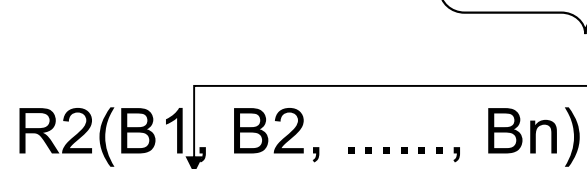
- ❑ Schéma de relation : intention de la relation
- ❑ Table : extension
- ❑ Schéma d'une BD relationnelle : ensemble des schémas des relations

## ■ Définition

- Une clé étrangère est un groupe d'attributs qui apparaît comme clé dans une autre relation

$R1(\underline{A1}, A2, \dots, \mathbf{Ap}, \mathbf{Ap+1}, \dots, An)$

$R2(\mathbf{B1}, B2, \dots, Bn)$



## ■ Rôle

- Les clés étrangères définissent des contraintes d'intégrités référentielles entre relations

# Clé étrangère (2)

- Mises à jour et clés étrangères
  - Insertion: la valeur des attributs doit exister dans la relation référencée.
    - Insertion de (4, 'BD', 15) dans Inscrit ?
  - Suppression dans la relation référencée; les n-uplets référençant doivent disparaître.
    - Suppression de l'élève 2 dans Elève ?
  - Les **clés étrangères** sont la **traduction** des **associations** du modèle E/A

# Clé étrangère

## ■ Exemples

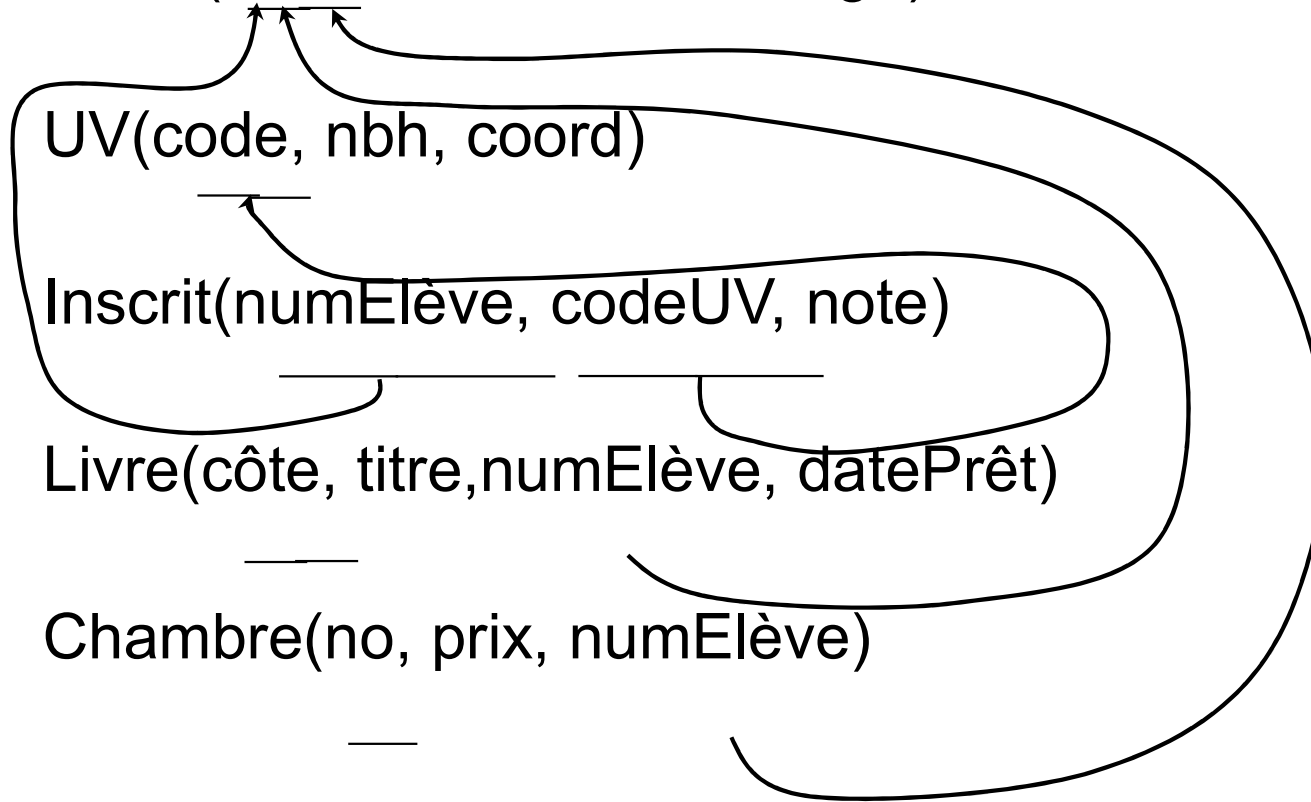
Élève(num, nom, adresse, age)

UV(code, nbh, coord)

Inscrit(numElève, codeUV, note)

Livre(côte, titre, numElève, datePrêt)

Chambre(no, prix, numElève)



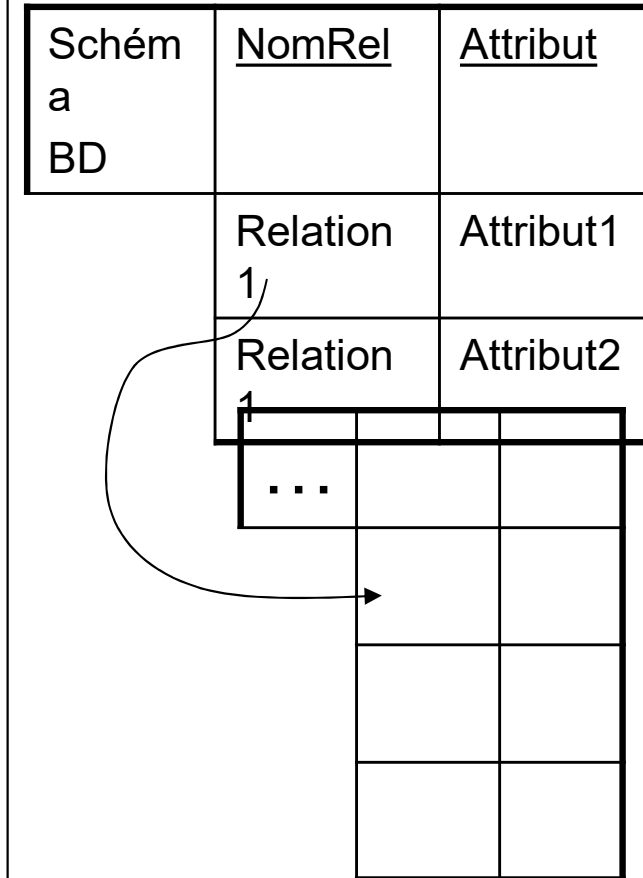
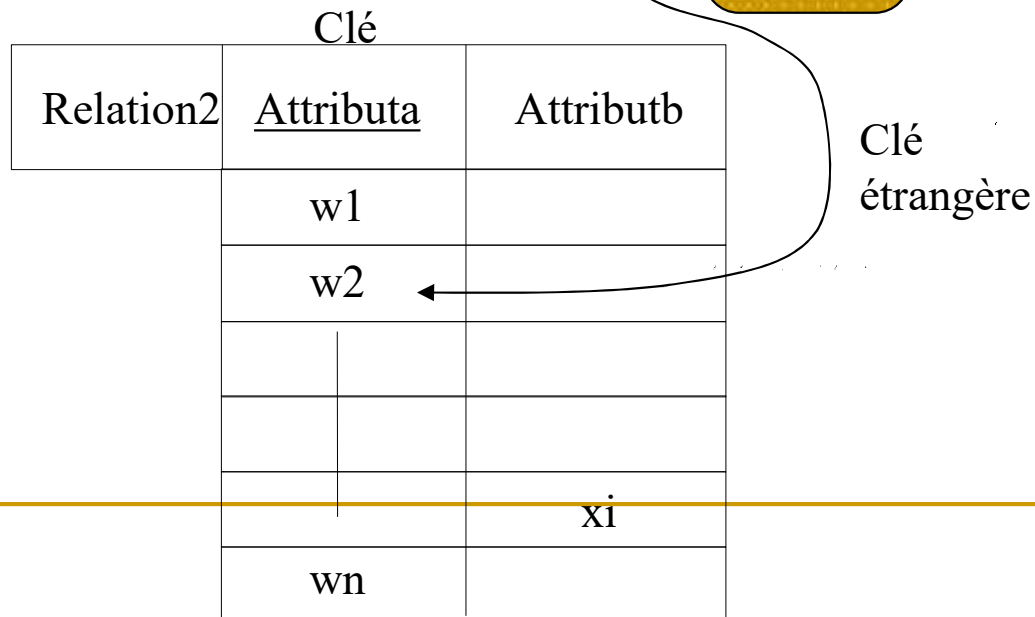
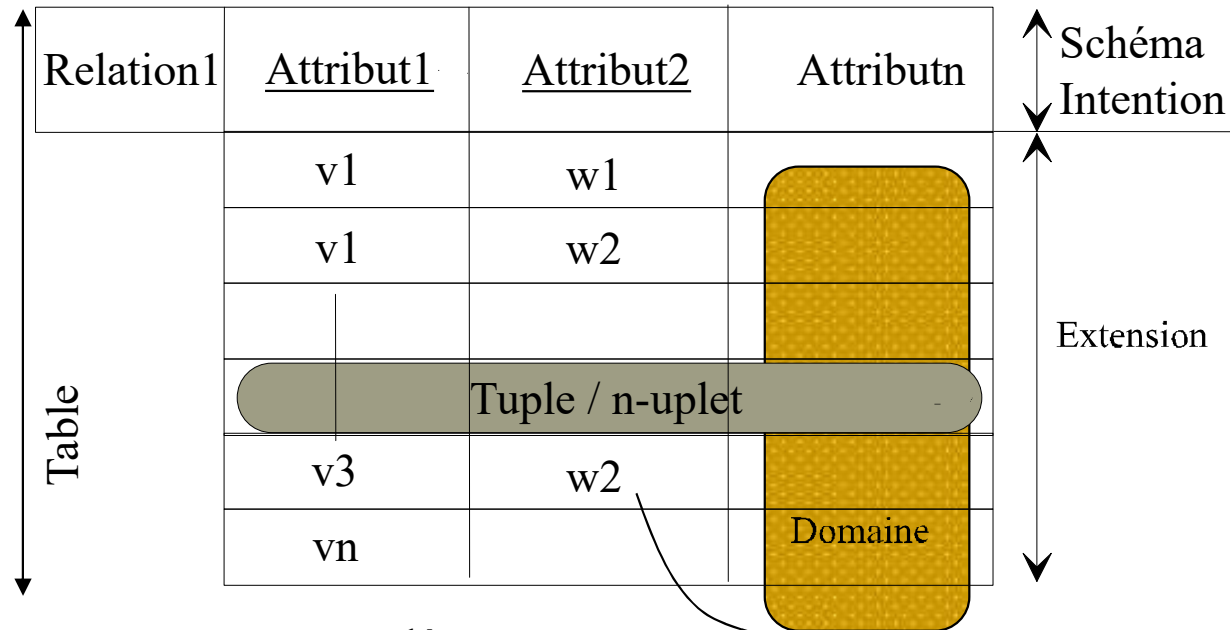
## ■ Définition

- base de données contenant l'ensemble des schémas et des règles de correspondances associées à une base de données

## ■ Principe

- Une base décrivant les autres bases, c'est-à-dire:
  - les relations
  - les attributs
  - les domaines
  - les clés .....
- Notion de dictionnaire de données
- Base particulière, système, gérée par l'administrateur de BD

# Résumé des notions



BD

Métabase

# Les dépendances fonctionnelles (DF)

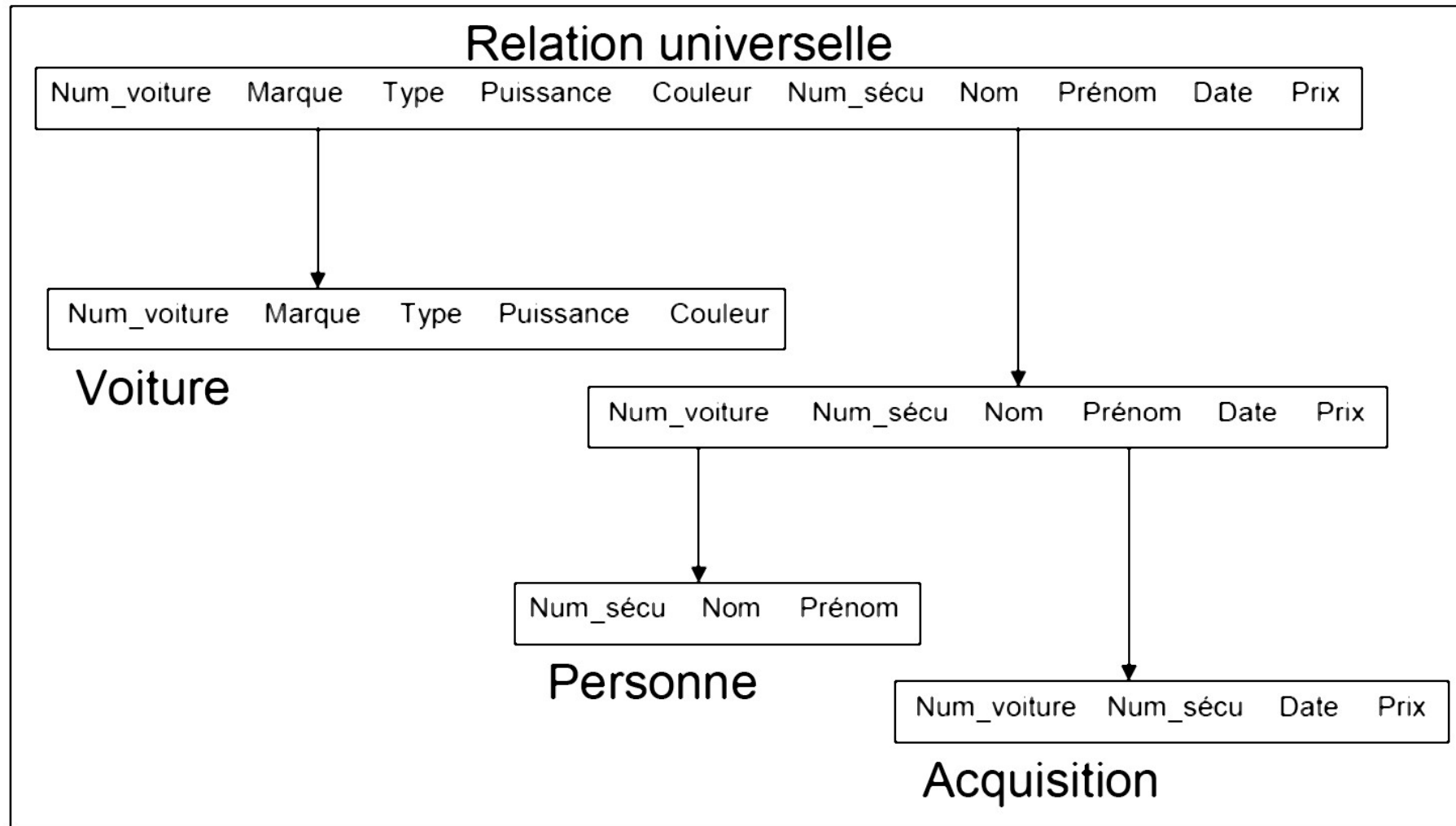
- Liens entre attributs qui ne sont pas explicités dans le schéma de la base.
- Permet de repérer :
  - –redondances
  - –anomalies
  - –contraintes
  - But : affiner, préciser, rendre plus concis un MCD



# Décomposition

- **Relation universelle** : relation unique dont le schéma est constitué de **tous** les attributs (les attributs du même concept ont le même nom, et vice versa).
- **Décomposition** : à partir de la relation universelle, isoler les entités et associations élémentaires (canoniques, ne pouvant être re-découpées) par un processus de raffinements successifs.

# Décomposition : exemple



# Décomposition et algèbre relationnel

## Décomposition:

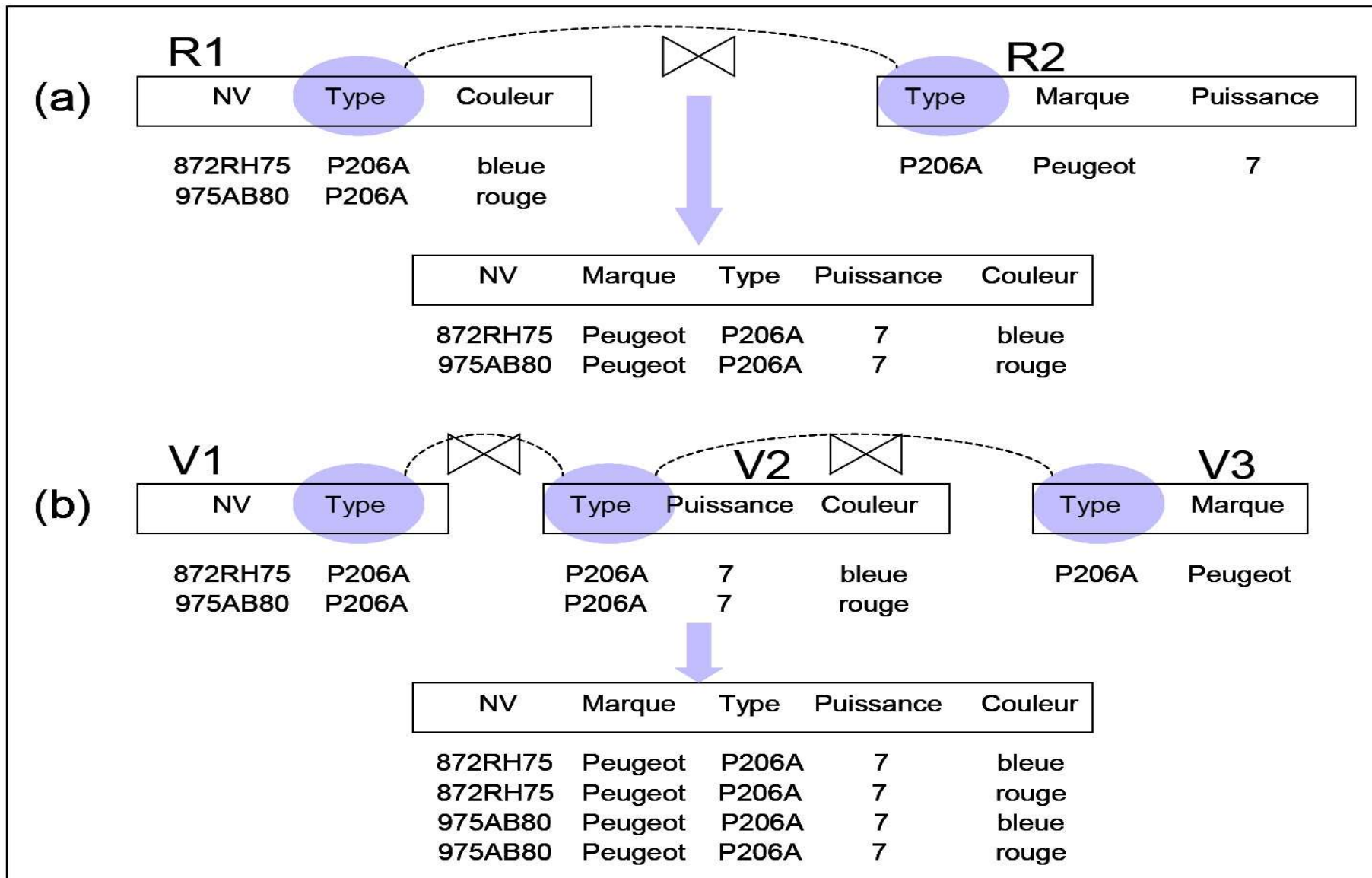
- Décomposition : projection
- Recomposition : jointure (naturelle) permet de retrouver la table initiale (si **décomposition sans perte**).
- En pratique, pas uniquement relation universelle : sur toute table “trop grosse”.
- Formellement : une décomposition d'une relation  $R(a_1, a_2, \dots, a_n)$  est son remplacement par une collection de relations  $R_1, R_2, \dots, R_m$  obtenues par projection sur  $R$  et telle que l'union de leur attributs contient tous les  $a_i$ . ie  $(a_1, a_2, \dots, a_n)$

# Perte d'information

- $R_1 \text{ NATURAL JOIN } R_2 \dots \text{ NATURAL JOIN } R_m$  a le même schéma que R mais pas forcément le même contenu :
  - -si même contenu : décomposition sans perte d'information
  - -décomposition avec perte sinon.
  - Exemple:

NV	Marque	Type	Puissance	Couleur
872RH75	Peugeot	P206A	7	bleue
975AB80	Peugeot	P206A	7	rouge

# Perte d'information : exemple



# Dépendance fonctionnelle et décomposition

- ● Les dépendances fonctionnelles permettent de sélectionner les attributs pour une décomposition sans perte.
- ● Ex.: marque, type et puissance partagent respectivement les mêmes valeurs : ces attributs sont fonctionnellement dépendants et donc il est intéressant que la décomposition les garde ensemble.

# Dépendance fonctionnelle

- DF: **lien sémantique non présent** dans le schéma de la base entre 2 attributs .
- Il peut être découvert par analyse des données (lignes) de la base.
- Formellement : Soit une relation  $R(a_1, a_2, \dots, a_n)$  et  $X$  et  $Y$  des sous-ensembles de  $\{a_1, a_2, \dots, a_n\}$ .
- On dit que  $X \rightarrow Y$  ( $Y$  **dépend fonctionnellement** de  $X$ ,  $X$  détermine  $Y$ )
- Autrement dit, si  $(x, y, z)$  et  $(x, y', z')$  sont deux tuples de  $R$ , alors  $y = y'$ .

# Propriétés des dépendances fonctionnelles

- **Réflexivité** : si  $Y \subseteq X$  alors  $X \rightarrow Y$ .
- **Augmentation** : si  $W \subseteq Z$  et  $X \rightarrow Y$  alors  $X, Z \rightarrow Y, W$ .
- **Transitivité** : si  $X \rightarrow Y$  et  $Y \rightarrow Z$  alors  $X \rightarrow Z$ .
- **Pseudo-transitivité** : si  $X \rightarrow Y$  et  $Y, Z \rightarrow T$  alors  $X, Z \rightarrow T$ .
- **Union** : si  $X \rightarrow Y$  et  $X \rightarrow Z$  alors  $X \rightarrow Y, Z$ .
- **Décomposition** : si  $Z \subseteq Y$  et  $X \rightarrow Y$  alors  $X \rightarrow Z$ .



# Définitions complémentaires

- La dépendance  $X \rightarrow Y$  est *élémentaire* s'il n'existe pas  $X' \subset X$  tel que  $X' \rightarrow Y$  (il n'y a pas d'attributs superflus dans la partie gauche de la dépendance).
- 
- La dépendance  $X \rightarrow Y$  est *directe* s'il n'existe pas  $Z$  dans  $R$  distinct de  $X$  et  $Y$  tel que  $X \rightarrow Z$  et  $Z \rightarrow Y$  (la dépendance n'est pas obtenue par transitivité).
- 
- La dépendance  $X \rightarrow Y$  est *triviale* si  $Y - X$  est vide.
- 
- Une dépendance fonctionnelle est *simple* si elle ne comporte qu'un seul attribut en partie droite et si elle n'est pas triviale.
- 
- $X \rightarrow A_1, A_2, \dots, A_n \Leftrightarrow \{ X \rightarrow A_1 ; X \rightarrow A_2 ; \dots ; X \rightarrow A_n \}$   
(en appliquant d'une façon itérative la propriété de décomposition).
- 
- Il est toujours possible de présenter les dépendances fonctionnelles sous forme simple.

# Fermeture et couverture d'un ensemble de DF

- On dira qu'un ensemble de dépendances  $D$  défini sur un ensemble de constituants  $X$  *implique logiquement* une dépendance  $d$ , ou que  $d$  est une *conséquence logique* de  $D$ , si l'ensemble de dépendances  $D$  étant vérifié sur  $X$ , alors  $d$  est aussi vérifiée sur  $X$ . On écrira  $D \rightarrow d$ .
- **La fermeture transitive**  $D^+$  de  $D$  est obtenue en inférant jusqu'à saturation les transitivités des DF sur l'ensemble initial  $D$  de dépendances.
- 2 ensembles de DFs sont **équivalents** si ils possèdent la même fermeture transitive.

# Couverture minimale: Définition 1

- **Couverture minimale** : plus petit ensemble de DFs équivalent à un ensemble de DFs. Contient donc les DFs qui ne peuvent être déduites des autres.
- Tout ensemble de dépendances fonctionnelles possède une couverture minimale (pas forcément unique) composée de dépendances fonctionnelles **dont les parties droites contiennent 1 seul attribut**.
- Notion très importante pour la décomposition des relations.

# Couverture minimale: Définition 2

- Couverture minimale ou *couverture irredondante* de  $D =$  sous-ensemble  $D^\circ$  de  $D$  tel que pour toute dépendance  $d \in D^\circ$ ,  $D^\circ - \{d\}$  n'implique pas  $d$ .
- Propriétés de la couverture minimale  $D^\circ$  de  $D$  :
  - les dépendances de  $D^\circ$  sont élémentaires et simples ;
  - la fermeture de  $D^\circ$  est égale à la fermeture de  $D$  ;
  - il n'existe pas de partie stricte  $D'$  de  $D^\circ$  dont la fermeture est égale à la fermeture de  $D$ .

# Algorithme de calcul de la couverture minimale

- Entrée : un ensemble de DF  $F$ ; Sortie : Couverture minimale de  $F$ 
  1.  $F' = F$
  2. Remplacer chaque dépendance fonctionnelle de la forme  $X \rightarrow (A1, \dots, An)$  dans  $F'$  par  $n$  dépendances fonctionnelles de la forme  $X \rightarrow A1, X \rightarrow A2, \dots, X \rightarrow An$  (cela revient à décomposer chaque dépendance fonctionnelle pour qu'elle n'ait plus qu'un seul attribut à droite).
    1. Pour chaque dépendance fonctionnelle  $X \rightarrow A$  dans  $F'$  : Pour chaque attribut  $B$  dans  $X$  :
      - Si  $(F' \setminus \{X \rightarrow A\}) \cup \{(X \setminus \{B\} \rightarrow A)$  est équivalent à  $F'$
      - Alors remplacer  $X \rightarrow A$  dans  $F'$  par  $(X \setminus \{B\}) \rightarrow A$
    1. Pour chaque dépendance fonctionnelle  $X \rightarrow A$  restant dans  $F'$  faire Si  $(F' \setminus \{X \rightarrow A\})$  est équivalent à  $F'$ 
      - Alors retirer  $X \rightarrow A$  de  $F'$  -- élimine les DF redondantes

# DF et clé primaire

## a) *Constituant et clés*

- *Constituant* = tout sous-ensemble de l'ensemble U des attributs d'une relation R.  
Le constituant Y est *totalelement dépendant* de X si la dépendance fonctionnelle
- $X \rightarrow Y$  est élémentaire.
- *Clé* = tout constituant X de R ( $A_1, A_2, \dots, A_n$ ) tel que :
  - $X \rightarrow A_1, A_2, \dots, A_n$  ;
  - il n'existe pas  $Y \subset X$  tel que  $Y \rightarrow A_1, A_2, \dots, A_n$ .
- *Constituant clé* = constituant qui appartient à l'une des clés candidates de R.
  - **Moins formellement** : une clé est un ensemble **minimal** d'attributs qui détermine tous les autres.
  - **Clé candidate** (à être primaire) : toute clé.

# Formes normales

- Permettent de faciliter (guider) la décomposition de relation sans perte d'information : conception
  - -précise
  - -cohérente
  - -sans redondance
- Plusieurs formes normales :
  - -les trois premières :-) pour la décomposition sans perte
  - -la forme normal de Boyce-Codd
  - -d'autres...

# La première forme normale

- Une relation est en première forme normale **si tout attribut contient une valeur atomique** (pas d'attribut multi-valué irrégulier).
- Autre formulation : il n'y a pas dans la relation d'attribut dont la valeur est, par exemple, une liste de valeurs.
- simple et esthétique.
- Facile à mettre en oeuvre : **transformer toute relation  $R(K,A)$  où  $A$  prend  $n$  valeurs en  $n$  relations  $R_i(K,A)$ .**
- Ex:  $\text{Personne}(\underline{\text{nom}}, \text{prénoms})$  en  $\text{Personne1}(\underline{\text{nom}}, \text{prenom1})$  et  $\text{Personne2}(\underline{\text{nom}}, \text{prenom2})$ . Ou  $\text{Personne}(\underline{\text{nom}})$  et  $\text{Prenom}(\underline{\text{nom}}, \underline{\text{prenom}})$



# Deuxième forme normale

- Une relation  $R(A, B, C, D, E)$  est en deuxième forme normale si et seulement si:
  - elle est en 1FN et:
  - tout constituant non clé est totalement dépendant de la clé de  $R$  (aucun des attributs  $C$ ,  $D$  et  $E$  ne dépend d'une partie de la clé).
- Autrement dit, il faut éviter la configuration suivante :  
 $A, B \rightarrow D, E ;$   
 $B \rightarrow C.$

# Troisième forme normale (3FN)

- But : éliminer les redondances liées aux dépendances fonctionnelles transitives.
- Une relation  $R(\underline{A}, \underline{B}, C, D, E)$  est en *troisième forme normale* si et seulement si elle est en 2FN et s'il n'existe aucune dépendance transitive entre une clé et un des attributs non clé (toutes les DF sont directes). Autrement dit,  $R$  est en 3FN s'il n'existe aucune dépendance entre deux attributs non clés. Comme  $R$  possède plusieurs clés, la définition doit être vérifiée pour chaque clé.
- Il faut donc éviter la configuration suivante :
- $A, B \rightarrow C, D, E ;$   
 $C \rightarrow E.$

# Préservation des DF

- Une décomposition d'une relation R **préservant les dépendances fonctionnelles** est  $(R_1, R_2 \dots R_n)$  telle que la fermeture transitive de R est égale à l'union des fermetures transitives des  $R_i$ .
  - 3FN : celle que l'on souhaite obtenir
  - Toute relation a au moins une décomposition en 3FN telle que :
    - préservation des dépendances fonctionnelles
    - décomposition sans perte

# Algorithme d'agrégation en 3FN d'un schéma

$R = \langle U, D \rangle$  :

- 1) Rechercher une couverture minimale  $G$  de  $D$ .
  - 2) Partitionner  $G$  en groupes de dépendances  $G_i$  ayant la même partie gauche.
  - 3) Fusionner les groupes  $G_i$  et  $G_j$  possédant des parties gauches  $X_i$  et  $X_j$  équivalentes (c'est-à-dire ceux pour lesquels on a  $X_i \rightarrow X_j$  et  $X_j \rightarrow X_i$ ,  $X_i \in G_i$ ,  $X_j \in G_j$ ).
  - 4) Associer à chaque groupe  $G_i$  un schéma  $R_i = \langle U_i, D_i \rangle$ . Autrement dit, construire une relation  $R_i$  pour chaque  $G_i$  dont la clé est la partie gauche des DF de  $D_i$  et les constituants non clés est la partie droite des DF de  $D_i$ .
  - 5) Si aucun des schémas précédents ne contient une clé  $K$  de  $R$ , créer un schéma supplémentaire  $\langle K, \emptyset \rangle$ .
- Les étapes 2 et 3 permettent de répartir toutes les dépendances fonctionnelles dans des schémas en 3FN. Cet algorithme assure donc bien la préservation des dépendances. La préservation du contenu est garantie par l'étape 5.

# Langages associés au modèle relationnel

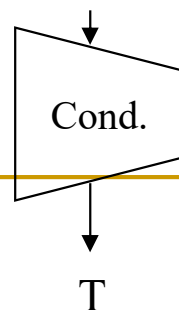
- Langages de Définition de Données (LDD) :
  - Définition /mise à jour des schémas des relations
- Langages de manipulation de données (LMD) :
  - Interrogation : recherche de données
  - Mises à jour : insertion, suppression, modification
- 2 classes de langages :
  - Algébriques → SQL
  - Prédicatifs → QBE

# Les opérateurs de manipulation

- Tout résultat d'une opération est une relation; peut donc être réutilisée en entrée d'un nouvel opérateur.
- Les opérateurs peuvent être classifiés en :
  - opérateurs ensemblistes / opérateurs relationnels
  - opérateurs de base / opérateurs dérivés
  - opérateurs unaires / opérateurs binaires
    - Unaires : sélection (restriction), projection,
    - Binaires : union, intersection, différence, produit cartésien, jointure, division

# Restriction

- But
  - Permet de "sélectionner" des tuples
  - La restriction réduit la taille de la relation verticalement
- Contraintes
  - Unaire
  - Spécifier une condition
- Notation
  - Notation textuelle:  
 $T \leftarrow \sigma_{\text{cond}}(R)$
  - Notation graphique:



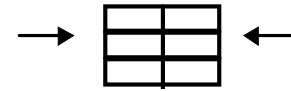
- Inscrits en BD :  
 $\sigma_{\text{codeUV}='BD'}(\text{Inscrit})$

Resu	NumElève	CodeUV	Note
	2	BD	10
	1	BD	20

- Majors (note > 15) de BD :  
 $\sigma_{\text{codeUV}='BD' \text{ et } \text{note} > 15}(\text{Inscrit})$

Resu	NumElève	CodeUV	Note
	1	BD	20

# Projection



## ■ But

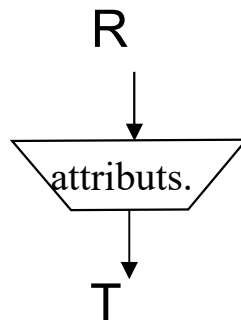
- ❑ Permet de "sélectionner" des attributs
- ❑ La projection réduit la taille de la relation horizontalement

## ■ Contraintes

- ❑ Unaire
- ❑ Spécifier une liste d'attributs

## ■ Notation

- ❑ Notation textuelle:  $T \leftarrow \Pi_{\text{attributs}}(R)$
- ❑ Notation graphique:



## ■ Adresses des élèves :

$\Pi_{\text{Adresse}}(\text{Elève})$

Resu	Adresse
.	Jijel
.	Bougie

Pas de doublon

## ■ Code et nb heures des UV :

$\Pi_{\text{code,nbh}}(\text{UV})$

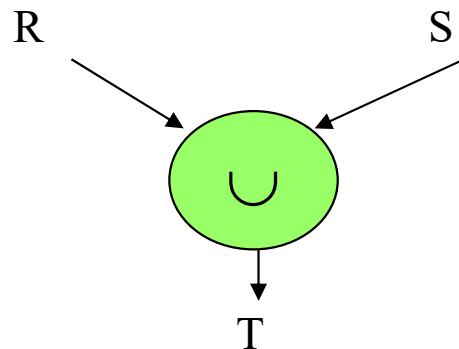
Resu	Code	nbh
	SI	45
	BD	15



# Union



- But
  - Permet de fusionner 2 relations
- Contraintes
  - Binaire
  - **Même schéma**
- Notation
  - Notation textuelle:  $T \leftarrow R \cup S$
  - Notation graphique:



## ■ Nom des profs, des élèves

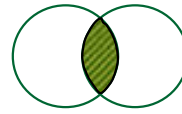
Prof	Nom	Elève2	Nom
	Univ-jijel		Bélaïd
	Univ-Bougie		Assil
	Assil		Taleb

- Nom des personnes à l'INT :  $\text{Prof} \cup \text{Eleve2}$

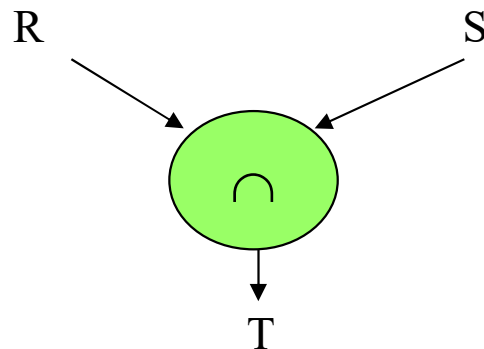
Resu	Nom
	Univ-jijel
	Univ-Bougie
	Assil
	Bélaïd
	Taleb

Pas de doublon

# Intersection



- But
  - Permet d'obtenir l'ensemble des tuples appartenant à deux relations
- Contraintes
  - Binaire
  - **Même schéma**
- Notation
  - Notation textuelle:  $T \leftarrow R \cap S$
  - Notation graphique:



- Nom des profs, des élèves

Prof	Nom	Elève2	Nom
	Univ-jijel		Bélaïd
	Univ-Bougie		Assil
	Assil		Taleb

- Noms communs élèves-profs :  $\text{Prof} \cap \text{Elève2}$

Resu	Nom
	Assil

# Différence

## ■ But

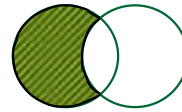
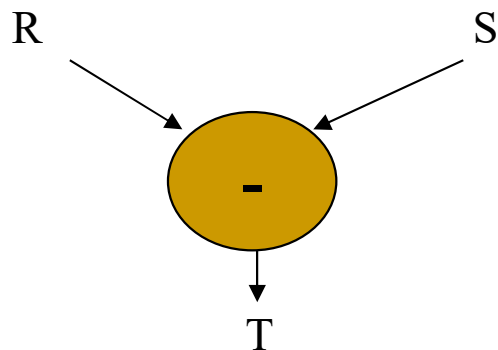
- Obtenir l'ensemble des tuples d'une relation qui ne figurent pas dans une autre

## ■ Contraintes

- Binaire
- **Même schéma**
- Non commutatif

## ■ Notation

- Notation textuelle:  $T \leftarrow R - S$
- Notation graphique:



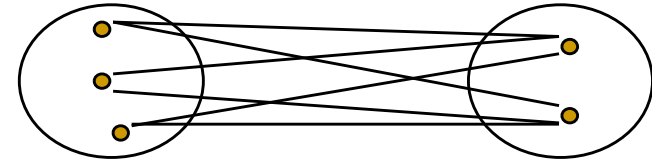
## ■ Nom des profs, des élèves

Prof	Nom	Elève2	Nom
	Univ-jijel		Bélaïd
	Univ-Bougie		Assil
	Assil		Taleb

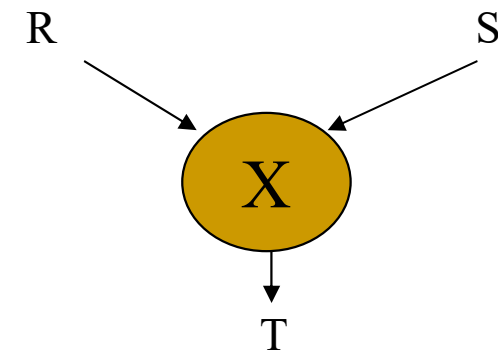
## ■ Noms des élèves qui ne portent pas le nom d'un prof : Eleve2-Prof

Résu	Nom
	Bélaïd
	Taleb

# Produit cartésien



- But
  - ❑ Ensemble de tous les tuples obtenus par concaténation de chaque tuple de R avec chaque tuple de S
- Contraintes
  - ❑ Binaire
  - ❑ Schéma du résultat:
    - $R(a_1, a_2, \dots, a_n), S(b_1, b_2, \dots, b_p)$
    - $T \leftarrow R \times S, T(a_1, a_2, \dots, a_n, b_1, b_2, \dots, b_p)$
  - ❑  $\text{Card}(R \times S) = \text{Card}(R) * \text{Card}(S)$
- Notation
  - ❑ Notation textuelle:  $T \leftarrow R \times S$
  - ❑ Notation graphique:

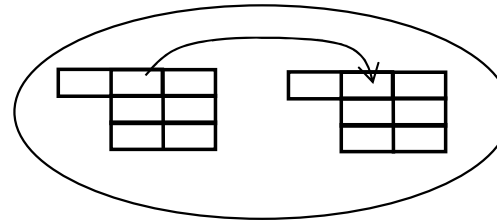


# Produit cartésien (2)

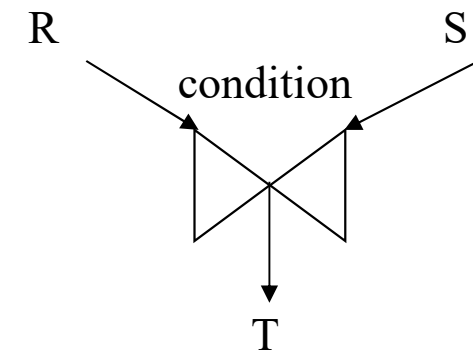
Elève	Num	Nom	Adresse	Age
	1	Bélaïd	Jijel	20
	2	Assil	Bougie	20
	3	Taleb	Jijel	21

UV	Code	Nbh	Coord
	SI	45	Univ-jijel
	BD	15	Univ-Bougie

Elève X UV	Num	Nom	Adresse	Age	Code	Nbh	Coord
	1	Bélaïd	Jijel	20	SI	45	Univ-jijel
	2	Assil	Bougie	20	SI	45	Univ-jijel
	3	Taleb	Jijel	21	SI	45	Univ-jijel
	1	Bélaïd	Jijel	20	BD	15	Univ-Bougie
	2	Assil	Bougie	20	BD	15	Univ-Bougie
	3	Taleb	Jijel	21	BD	15	Univ-Bougie



- But
  - ❑ Permet d'établir le lien sémantique entre les relations
- Contraintes
  - ❑ Binaire
  - ❑ Schéma du résultat:
    - $R(a_1, a_2, \dots, a_n), S(b_1, b_2, \dots, b_p)$
    - $T \leftarrow R \bowtie_{\text{condition}} S \quad T(a_1, a_2, \dots, a_n, b_1, b_2, \dots, b_p)$
- Notation
  - ❑ Notation textuelle:  $T \leftarrow R \bowtie_{\text{condition}} S$
  - ❑ Notation graphique:



# 1er exemple de jointure

Elève	<u>Num</u>	Nom	Adresse	Age
	1	Bélaïd	Jijel	20
	2	Assil	Bougie	20
	3	Taleb	Jijel	21

Elève.Num=Chambre.numElève

Chambre	<u>No</u>	Prix	numElève
	10	200	3
	21	150	2

Elève ▷◁Chambre	<u>Num</u>	Nom	Adresse	Age	No	Prix	numElève
	2	Assil	Bougie	20	21	150	2
	3	Taleb	Jijel	21	10	200	3

- 1 tuple de Chambre → 1 tuple de résultat
- 1 tuple de Elève → **0** ou 1 tuple de résultat
  - On a perdu Bélaïd !

## 2ème exemple de jointure

Inscrit	<u>NumElève</u>	<u>CodeUV</u>	Note
	2	BD	10
	1	BD	20
	2	SI	17
	3	SI	18

Inscrit.NumElève=Elève.Num

Elève	<u>Num</u>	Nom	Adresse	Age
	1	Bélaïd	Jijel	20
	2	Assil	Bougie	20
	3	Taleb	Jijel	21

Inscrit ▷◁Elève	<u>Num</u>	Nom	Adresse	Age	<u>NumElève</u>	<u>CodeUV</u>	Note
	1	Bélaïd	Jijel	20	1	BD	20
	2	Assil	Bougie	20	2	SI	17
	2	Assil	Bougie	20	2	BD	10
	3	Taleb	Jijel	21	3	SI	18

- 1 tuple de Inscrit → 1 tuple de résultat
- 1 tuple de Elève → 0 à n tuples de résultat
  - On a dupliqué Assil !



### ■ But

- Répondre aux requêtes de type « tous les »
- Un tuple  $t$  est dans  $T$  si et seulement si pour tout tuple  $s$  de  $S$ , le tuple  $\langle t, s \rangle$  est dans  $R$

### ■ Contraintes

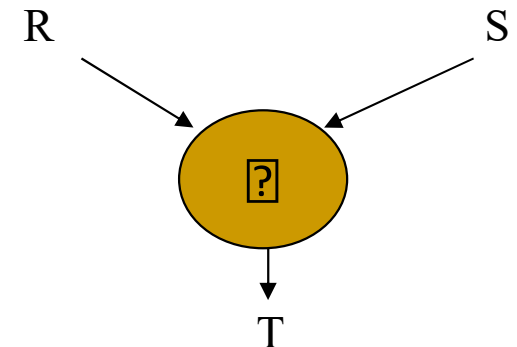
- Binaire
- Schéma du résultat:
  - $R(a_1, a_2, \dots, a_n, b_1, b_2, \dots, b_p), S(b_1, b_2, \dots, b_p)$
  - $T \leftarrow R \div S, T(a_1, a_2, \dots, a_n)$

### ■ Notation

- Notation textuelle:  $T = R \text{ ? } S$
- Notation graphique:

### ■ Dérivation

- Projection + Produit cartésien + Différence.
- $R \text{ ? } S \leftarrow T_1 - T_2$  avec:
  - $T_1 \leftarrow \Pi_{\text{schéma}(R) - \text{schéma}(S)}(R)$
  - $T_2 \leftarrow \Pi_{\text{schéma}(R) - \text{schéma}(S)}((\Pi_{\text{schéma}(R) - \text{schéma}(S)}(R) \times S) - R)$



# Division (2)

## ■ Exemple

- Quels sont les élèves inscrits à toutes les UVs ?

Inscrit	NumElève	CodeUV	Note
	2	BD	10
	1	BD	20
	2	SI	17
	3	SI	18

# Division (3)

## ■ Exemple

- ❑ Construire R : ensemble de toutes les informations dont on a besoin = attributs NumElève et CodeUV de Inscrit (R)
- ❑ Construire S : ensemble correspondant à "tous les" (UV) = codeUV (S)
- ❑ Résultat  $\leftarrow R \div S$
- ❑ Vérification :
  - Résultat  $\times S \subseteq R$

- ❑  $R \leftarrow \Pi_{\text{NumElève, CodeUV}}(\text{Inscrit})$

R	NumElève	CodeUV
	2	BD
	1	BD
	2	SI
	3	SI


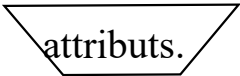




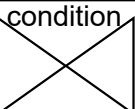

- ❑  $S \leftarrow \Pi_{\text{Code}}(\text{UV})$

S	CodeUV
	BD
	SI

- ❑ Résultat

Resu	NumElève
	2

# Bilan : sémantique et notations des opérateurs

Opérateur	Sémantique	Notation textuelle	Notation graphique
Restriction	« Sélectionner » des tuples	$T \leftarrow \sigma_{\text{cond}}(R)$	
Projection	« Sélectionner » des attributs	$T \leftarrow \Pi_{\text{attributs}}(R)$	
Union	Fusionner les extensions de 2 relations	$T \leftarrow R \cup S$	
Intersection	Obtenir l'ensemble des tuples communs à deux relations	$T \leftarrow R \cap S$	
Différence	Tuples d'une relation qui ne figurent pas dans une autre	$T \leftarrow R - S$	
Produit cartésien	Concaténer chaque tuple de R avec chaque tuple de S	$T \leftarrow R \times S$	
Jointure	Etablir le lien sémantique entre les relations	$T \leftarrow R \bowtie_{\text{condition}} S$	
Division	Répondre aux requêtes de type « tous les »	$T \leftarrow R \oslash S$	

# Bilan : contraintes des opérateurs

Opérateur	Unaire/ Binaire	Schémas	« Paramètres »
Restriction $T \leftarrow \sigma_{\text{cond}}(R)$	Unaire	$\text{Schéma}(T) = \text{Schéma}(R)$	Condition sur attributs de R
Projection $T \leftarrow \Pi_{\text{attributs}}(R)$	Unaire	$\text{Schéma}(T) \subseteq \text{Schéma}(R)$	Liste d'attributs de R
Union $T \leftarrow R \cup S$	Binaire	$\text{Schéma}(R) = \text{Schéma}(S) = \text{Schéma}(T)$	
Intersection $T \leftarrow R \cap S$	Binaire	$\text{Schéma}(R) = \text{Schéma}(S) = \text{Schéma}(T)$	
Différence $T \leftarrow R - S$	Binaire	$\text{Schéma}(R) = \text{Schéma}(S) = \text{Schéma}(T)$	
Produit cartésien $T \leftarrow R \times S$	Binaire	$\text{Schéma}(T) = \text{Schéma}(S) \cup \text{Schéma}(R)$	
Jointure $T \leftarrow R \bowtie_{\text{condition}} S$	Binaire	$\text{Schéma}(T) = \text{Schéma}(S) \cup \text{Schéma}(R)$	Condition de jointure sur attributs de R et S
Division $T \leftarrow R \div S$	Binaire	$\text{Schéma}(R) = \text{Schéma}(S) + \text{Schéma}(T)$	

# Exemples de requêtes en algèbre relationnelle

Exemples

## ■ Base de données exemple : les vins

Produits(num\_P, Designation, annee, Prix\_Unit)

P

Depots(Num\_p, num\_F, quantite)

D

Fournisseurs(num\_F, nom, prenom, region)

F

Clients(num\_C, nom, prenom, ville)

Cl

Commandes(ncde, num\_C, date, Num\_P, qte)

C

# Composition des opérateurs

## Exemples

- Noms des produits qui coutent 1000 DA,

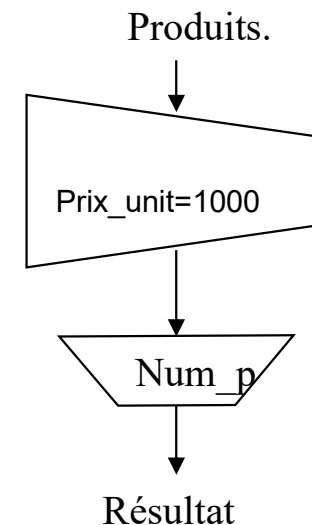
- Ecriture textuelle :

$\text{Temp} \leftarrow \sigma_{\text{Prix\_unit}=1000}(\text{Produits})$

$\text{Resultat} \leftarrow \Pi_{\text{num\_p}}(\text{Temp})$

$\text{Resultat} \leftarrow \Pi_{\text{num\_p}}(\sigma_{\text{Prix\_unit}=1000}(\text{Produits}))$

- Arbre algébrique :



# Noms des Fournisseurs des stylos

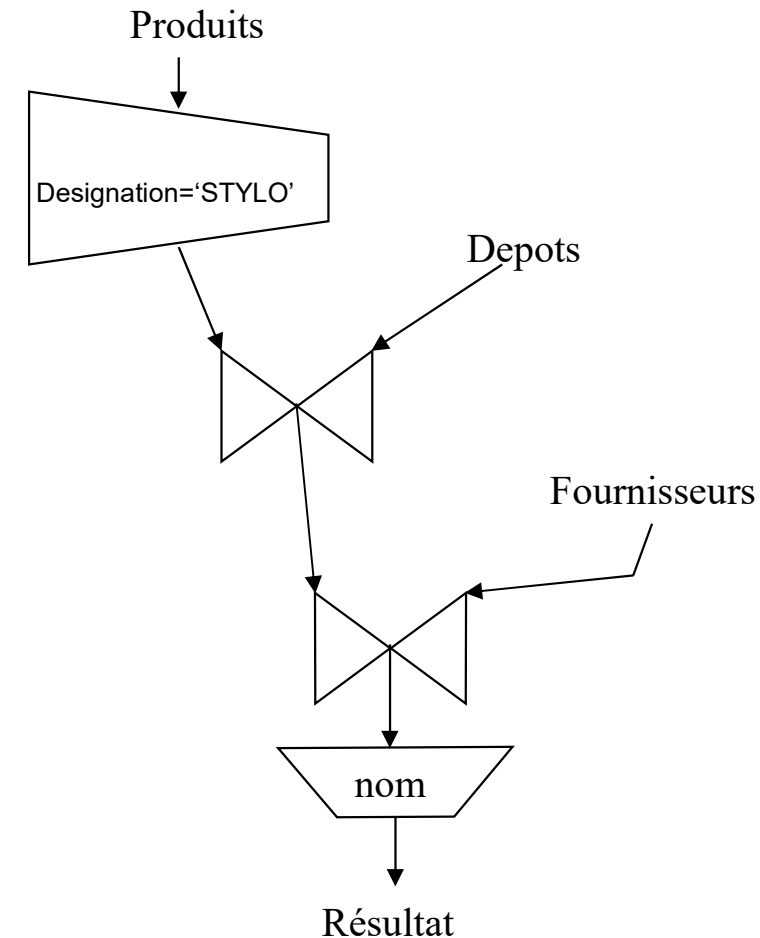
## Exemples

$\text{Res1} \leftarrow \sigma_{\text{Designation}='STYLO'} (P)$

$\text{Res2} \leftarrow \text{Res1} \bowtie D$

$\text{Res3} \leftarrow \text{Res2} \bowtie F$

$\text{Resultat} \leftarrow \Pi_{\text{nom}}(\text{Res3})$





# Numéros des Produits ne faisant l'objet d'aucune commande

Exemples

$$\text{Resultat} \leftarrow \Pi_{\text{num\_P}}(P) - \Pi_{\text{Num\_p}}(C)$$

