

# INFORMATIQUE Développement d'applications

## BLOC 1

### UE05 Bases de données

## Chapitre 2 : Le modèle relationnel

Vincent Reip

Octobre 2024

# Objectif

- Au terme de ce chapitre, l'étudiant sera capable de :
  - comprendre les concepts et termes essentiels liés au modèle relationnel et aux transformations de relations via l'algèbre relationnelle.
  - comprendre les règles de normalisation d'un schéma relationnel

# Le modèle relationnel ?

- Le modèle relationnel modélise les relations existant entre des données (proposé par **Edgar F. Codd** (chercheur chez IBM) en 1970 )
- Un relation est un ensemble d'attributs qui caractérisent un objet
  - Ex : la relation Etudiant ➔ un étudiant a un matricule, un nom, un prénom et une adresse email (« Q220987 », « Dupont », « Roger », « rogerd@dupont.be »)
- Les relations sont représentées sous formes de **tables** qui modélisent une entité (un objet du monde réel)
  - Une **ligne (tuple)** représente une **occurrence** de l'entité
  - Une **colonne (attribut)** représente une **propriété** de l'entité

# Le modèle relationnel

- Le modèle relationnel repose sur les principes mathématiques de la théorie des ensembles
- Ce modèle propose une organisation des données simple, dégagée de toute considération technologique
- Il est accompagné d'un ensemble d'opérations relationnelles permettant de manipuler les données : l'algèbre relationnelle
  - Opérateurs qui permettent de construire de nouvelles relations à partir d'autres relations
  - ➔ base du langage SQL

# Le modèle relationnel

- La représentation de données complexes peut mener à des bases de données de plusieurs centaines de tables
- Il nous faut donc des **règles** qui nous aideront à **construire** de telles bases de données et à **assurer la cohérence des données** au sein de celles-ci
  - Définitions
  - Contraintes
  - Formes normales

# Objectifs du modèle relationnel

- Assurer l'**indépendance** entre les applications utilisant les données et la représentation physique de ces données
  - pas de notion de fichier, chemin d'accès, index... (niveau interne vs. niveau conceptuel)
- Fournir des moyens permettant de gérer la problématique de la **cohérence** et de la **redondance** des données
  - contraintes et normalisation
- Permettre la **manipulation** des données grâce à un langage de haut niveau
  - langage standardisé (SQL) basé sur l'algèbre relationnelle



# Définitions : Domaine

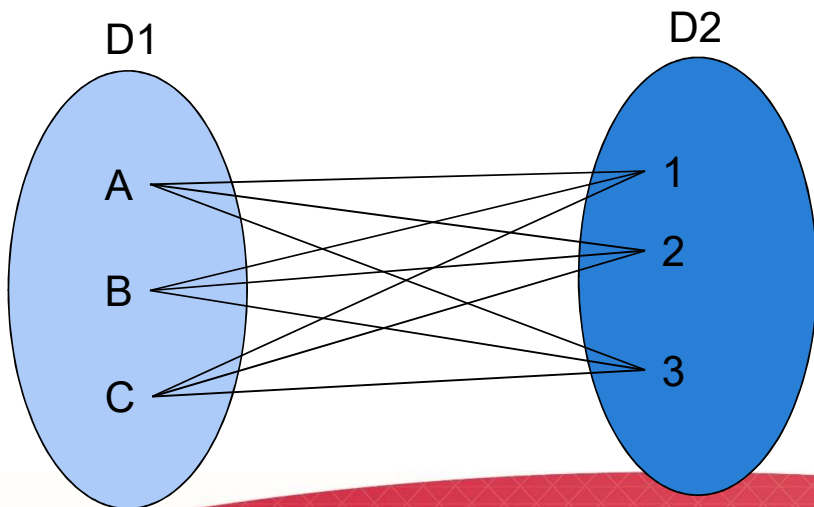
- **Domaine** : un domaine est un ensemble de valeurs
  - $N = \{0, 1, 2, 3, 4, \dots\}$  est le domaine des entiers
  - $\{0, 1\}$  est le domaine des booléens
  - $\{\text{Joe}, \text{Jack}, \text{William}, \text{Averell}\}$  est le domaine des prénoms des Dalton
  - $\{\text{Rouge}, \text{Orange}, \text{Jaune}, \text{Vert}, \text{Bleu}, \text{Indigo}, \text{Violet}\}$  est le domaine des couleurs de l'arc-en-ciel
  - ...



Dans une table (relation), chaque élément a une valeur appartenant à un domaine.

# Définitions : Produit cartésien

- **Produit cartésien** : le produit cartésien d'un ensemble de domaines  $D_1, D_2, \dots, D_n$  est l'ensemble des **tuples**  $\{v_1, v_2, \dots, v_n\}$  tels que  $v_i$  appartient à  $D_i$
- Le produit cartésien est noté  $D_1 \times D_2 \times \dots \times D_n$



$D_1 \times D_2$

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



Un produit cartésien présente rarement un intérêt dans le monde des bases de données.



# Définitions : Produit cartésien

- Soient les domaines suivants :
  - $D1 = \{\text{Roger, Marc, Jules, Simone, Natacha}\}$  le domaine des prénoms de mes frères et sœurs
  - $D2 = \{1, 2, 3, 4, \dots, 129, 130\}$  le domaine des âges plausibles
- Le produit cartésien de ces deux domaines associera les 130 valeurs du domaine D2 à chacune des 5 valeurs du domaine D1
  - 650 combinaisons...
  - ... dont 645 ne correspondent à aucune information réelle

# Définitions : Relation

- **Relation** : une relation désigne un **sous-ensemble nommé** du produit cartésien d'un ensemble déterminé de domaines.
- Chaque tuple d'un produit cartésien étant unique, chaque tuple d'une relation est lui aussi **unique**
- Sur base des domaines D1 et D2 (voir exemple précédent), on peut construire la relation suivante :

AGE\_FRATRIE

Roger	25
Marc	28
Jules	32
Simone	32
Natacha	35



Cette relation est porteuse d'information pertinente et potentiellement utile.

Une relation peut être représentée par une table dont les colonnes correspondent aux domaines de valeurs. Les lignes, contenant une valeur pour chaque domaine, sont appelées tuples.

# Définitions : Attribut

- **Attribut**
  - Identificateur (nom) décrivant un ensemble de valeurs de même type correspondant à une propriété d'une entité
  - On parle aussi de 'colonne' de la table
  - Ex : attribut AGE (type entier), attribut PRENOM (type chaîne de caractères)

AGE\_FRATRIE

PRENOM	AGE
Roger	25
Marc	28
Jules	32
Simone	32
Natacha	35

# Définitions : Schéma de relation

- Schéma de relation

- Un schéma de relation précise :
  - Le nom de la relation
  - La liste des attributs de la relation
  - Le type de chacun des attributs
  - Les éventuelles contraintes sur ces attributs

Pas d'importance de l'ordre

AGE_FRATRIE		
IDFRATRIE	Entier	Unique et obligatoire
PRENOM	Chaîne de caractères	Obligatoire
AGE	Entier	Compris entre 1 et 130

AGE\_FRATRIE = {  
 IDFRATRIE: Entier  
 PRENOM: Chaîne de caractères,  
 AGE: Entier  
 }

# Définitions : Tuple

- Tuple
  - Un tuple (ou n-uplet) correspond à une occurrence de la relation (une ligne de la table)
  - Au sein d'une relation, chaque tuple est unique

AGE\_FRATRIE

IDFRATRIE	PRENOM	AGE
1	Roger	25
2	Marc	28
3	Jules	32
4	Simone	32
5	Natacha	35



Comment assurer  
cette unicité des  
tuples ?

# Définitions : Degré et cardinalité

- Degré d'une relation (ou arité)
  - Nombre d'attributs qui composent la relation
- Cardinalité d'une relation
  - Nombre de tuples (d'occurrences) de la relation

AGE\_FRATRIE

IDFRATRIE	PRENOM	AGE
1	Roger	25
2	Marc	28
3	Jules	32
4	Simone	32
5	Natacha	35

Degré = 3

Cardinalité = 5



# Définitions : Clé candidate

- Une **clé candidate** d'une relation R est un sous-ensemble d'attributs X tels que :
  - 2 tuples de R ne peuvent avoir la même valeur pour l'ensemble des attributs de X (**contrainte d'unicité**)
  - Il n'existe pas de sous-ensemble de X respectant la contrainte d'unicité (**contrainte d'irréductibilité**).



En d'autres termes, une clé candidate aura une valeur unique pour chaque tuple de la relation et cette clé candidate est la plus petite possible (on élimine les attributs qui ne sont pas nécessaires pour assurer l'unicité).

# Définitions : Clé candidate

- Quelles sont les **clés candidates** de cette relation ?

EMPLOYE

MATRICULE	NOM	PRENOM	DDN	REGNAT	NUMGSM	RUE	CODEPOSTAL	LOCALITE
A1004	Dupont	Roger	25-01-1980	80.01.25-005.23	0474/98.45.78	Rue Armand Clair, 5	4000	Liège
A9098	Meert	Marc	28-06-1975	75.06.28-048.08	0495/78.45.91	Rue du tir, 15	4020	Liège
C5674	Lambert	Jules	12-05-1991	91.05.12-098.19	0475/16.23.25	Avenue des platanes, 56	5000	Namur
D0023	Siroin	Simone	01-02-1998	98.02.01-012.37	0476/48.98.98	Allée des roses, 156	4800	Verviers
C7845	Palout	Natacha	02-12-1987	87.12.02-005.76	0475/14.15.16	Rue Célestin Burion, 4	4000	Liège

# Définitions : Clé primaire

- Une **clé primaire** (Primary key) est choisie parmi les clés candidates pour jouer « officiellement » le rôle d'**identifiant** pour chacun des tuples.
- Une clé primaire ne peut être composée que d'attributs obligatoires (**intégrité d'entité**)
- Elle peut être choisie pour sa représentativité ou pour sa facilité de manipulation
- Il est fréquent d'**ajouter** un **attribut spécifique** à une relation dans le seul but de lui faire jouer le rôle de clé primaire
- Le ou les attribut(s) qui forme(nt) la clé primaire sont **souligné(s)**

# Définitions : Clé primaire

AGE\_FRATRIE

IDFRATRIE	PRENOM	AGE
1	Roger	25
2	Marc	28
3	Jules	32
4	Simone	32
5	Natacha	35

ETUDIANT
<u>Matricule</u>
Nom
Prenom
RegNat

```
CREATE TABLE ETUDIANT(
  Matricule CHAR(8),
  Nom VARCHAR(50) NOT NULL,
  Prenom VARCHAR(50) NOT NULL,
  RegNat CHAR(15) NOT NULL,
  PRIMARY KEY(Matricule),
  UNIQUE(RegNat)
);
```

EMPLOYE

MATRICULE	NOM	PRENOM	DDN	REGNAT	NUMGSM	RUE	CODEPOSTAL	LOCALITE
A1004	Dupont	Roger	25-01-1980	80.01.25-005.23	0474/98.45.78	Rue Armand Clair, 5	4000	Liège
A9098	Meert	Marc	28-06-1975	75.06.28-048.08	0495/78.45.91	Rue du tir, 15	4020	Liège
C5674	Lambert	Jules	12-05-1991	91.05.12-098.19	0475/16.23.25	Avenue des platanes, 56	5000	Namur
D0023	Siroin	Simone	01-02-1998	98.02.01-012.37	0476/48.98.98	Allée des roses, 156	4800	Verviers
C7845	Palout	Natacha	02-12-1987	87.12.02-005.76	0475/14.15.16	Rue Célestin Burion, 4	4000	Liège

# Définitions : Surclé

- Une **surclé** d'une relation R est un sous-ensemble des attributs de R qui respecte la contrainte d'unicité mais pas obligatoirement celle d'irréductibilité



Cette notion sera utile plus tard pour la définition des formes normales.

# Définition : Surclé

- Quelles sont les **surclés** de cette relation ?

EMPLOYE

MATRICULE	NOM	PRENOM	DDN	REGNAT	NUMGSM	RUE	CODEPOSTAL	LOCALITE
A1004	Dupont	Roger	25-01-1980	80.01.25-005.23	0474/98.45.78	Rue Armand Clair, 5	4000	Liège
A9098	Meert	Marc	28-06-1975	75.06.28-048.08	0495/78.45.91	Rue du tir, 15	4020	Liège
C5674	Lambert	Jules	12-05-1991	91.05.12-098.19	0475/16.23.25	Avenue des platanes, 56	5000	Namur
D0023	Siroin	Simone	01-02-1998	98.02.01-012.37	0476/48.98.98	Allée des roses, 156	4800	Verviers
C7845	Palout	Natacha	02-12-1987	87.12.02-005.76	0475/14.15.16	Rue Célestin Burion, 4	4000	Liège



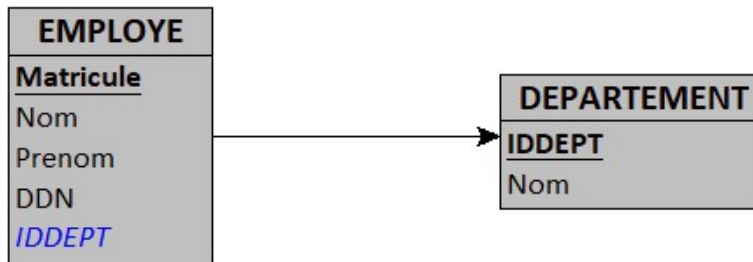
# Définitions : Clé étrangère

- Une **clé étrangère (Foreign key)** d'une relation R1 est un sous-ensemble d'attributs qui identifie un tuple d'une autre relation R2
- La clé étrangère de R1 **référence** une clé de R2, généralement la clé primaire (**identifiant cible**)



Les clés étrangères sont omniprésentes dans un modèle relationnel car elle permettent d'établir des liens entre des tuples de différentes relations.

# Définitions : Clé étrangère



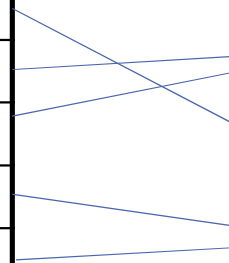
L'attribut IDDEPT de la table EMPLOYE est un clé étrangère (FK) qui référence la clé primaire (PK) IDDEPT de la table DEPARTEMENT.

EMPLOYE

<u>MATRICULE</u>	NOM	PRENOM	DDN	IDDEPT
A1004	Dupont	Roger	25-01-1980	2
A9098	Meert	Marc	28-06-1975	1
C5674	Lambert	Jules	12-05-1991	1
D0023	Siroin	Simone	01-02-1998	4
C7845	Palout	Natacha	02-12-1987	4

DEPARTEMENT

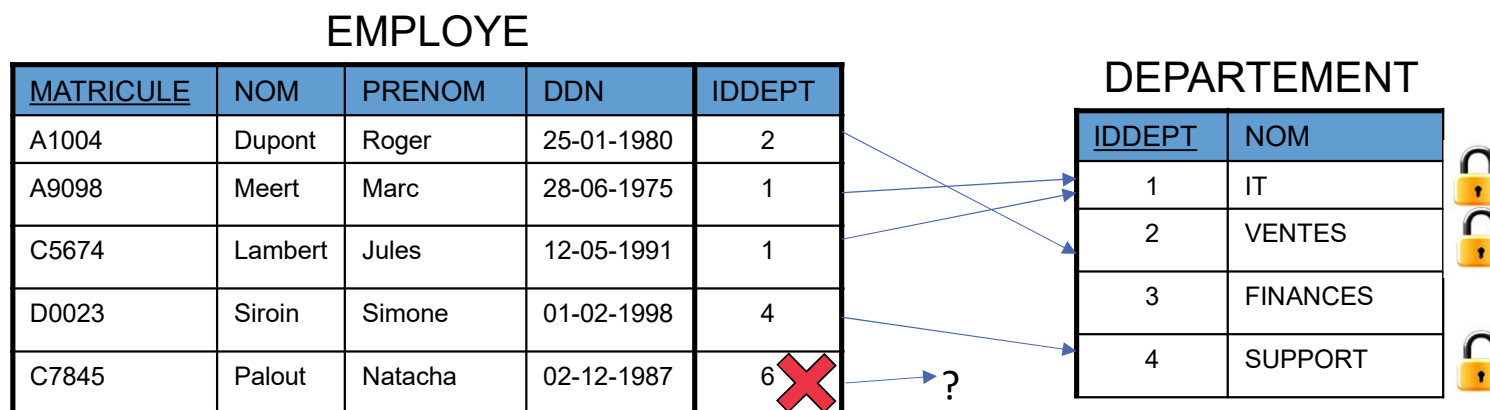
<u>IDDEPT</u>	NOM
1	IT
2	VENTES
3	FINANCES
4	SUPPORT



# Définitions : Contrainte d'intégrité référentielle

- **Contrainte d'intégrité référentielle**
  - L'ensemble des valeurs d'une **clé étrangère** doit être, à tout instant, un sous-ensemble des valeurs de l'identifiant cible
  - Cette contrainte est garantie par le SGBD

# Définitions : Contrainte d'intégrité référentielle



 On ne peut pas avoir un tuple dans la table EMPLOYE qui référence un tuple de la table DEPARTEMENT qui n'existe pas.



Si un tuple de la table DEPARTEMENT est référencé, on ne peut pas le supprimer.

# Définitions : contrainte d'unicité

- **Contrainte d'unicité**
  - Une contrainte d'unicité peut être définie sur un sous-ensemble des attributs d'une relation R
  - Cette contrainte assure que 2 tuples de R ne peuvent avoir les mêmes valeurs pour les attributs concernés
  - Cette contrainte est inhérente aux clés primaires ou candidates
  - L'utilisation de telles contraintes permet d'augmenter la cohérence des données.

# Définitions : contrainte d'unicité

UTILISATEUR

IDUTILISATEUR	NOM	PRENOM	EMAIL
1	Dupont	Roger	<a href="mailto:roger@gmail.com">roger@gmail.com</a>
2	Meert	Marc	<a href="mailto:mmeert@yahoo.be">mmeert@yahoo.be</a>
3	Lambert	Jules	<a href="mailto:jules48@yahoo.be">jules48@yahoo.be</a>
4	Siroin	Simone	<a href="mailto:Simsir_67@gmail.com">Simsir_67@gmail.com</a>
5	Palout	Natacha	<a href="mailto:natou67@yahoo.fr">natou67@yahoo.fr</a>



On pourrait définir un contrainte d'unicité sur l'attribut EMAIL de la table UTILISATEUR

DEPARTEMENT

IDDEPT	NOM
1	IT
2	VENTES
3	FINANCES
4	SUPPORT



On pourrait définir un contrainte d'unicité sur l'attribut NOM de la table DEPARTEMENT



# Définitions : contrainte NOT NULL

- **Attribut facultatif / obligatoire**
  - Facultatif : on peut admettre que pour certains tuples, la valeur d'un attribut ne soit pas connue
  - L'absence de valeur pour un attribut d'un tuple est signalée par le symbole **NULL**
  - Le SGBD garanti le caractère obligatoire d'un attribut si la contrainte « **NOT NULL** » a été spécifiée lors de la définition de l'attribut

EMPLOYE

MATRICULE	NOM	PRENOM	DDN
A1004	Dupont	Roger	25-01-1980
A9098	Meert	Marc	NULL
C5674	Lambert	Jules	12-05-1991
D0023	Siroin	Simone	NULL
C7845	Palout	Natacha	02-12-1987

# Types primitifs des attributs

- Type **integer**
  - Admet des nombres entiers
- Type **numeric(precision[, scale])**
  - Admet des nombres réels
  - **precision** indique le nombre de chiffres
  - **scale** (optionnel) mentionne le nombre de chiffres décimaux
    - Ex : numeric (5,2) représente un nombre de 5 chiffres dont 2 après la virgule.

# Types primitifs des attributs

- Type `char(longueur)`
  - Désigne des chaînes de caractères dont la longueur maximale est fixée par le paramètre '*longueur*' et qui occupent un espace fixe dans la base de données
- Type `varchar(longueur)`
  - Même utilité que char mais les chaînes occupent un espace variable d'octets dans la base de données

# Types primitifs des attributs

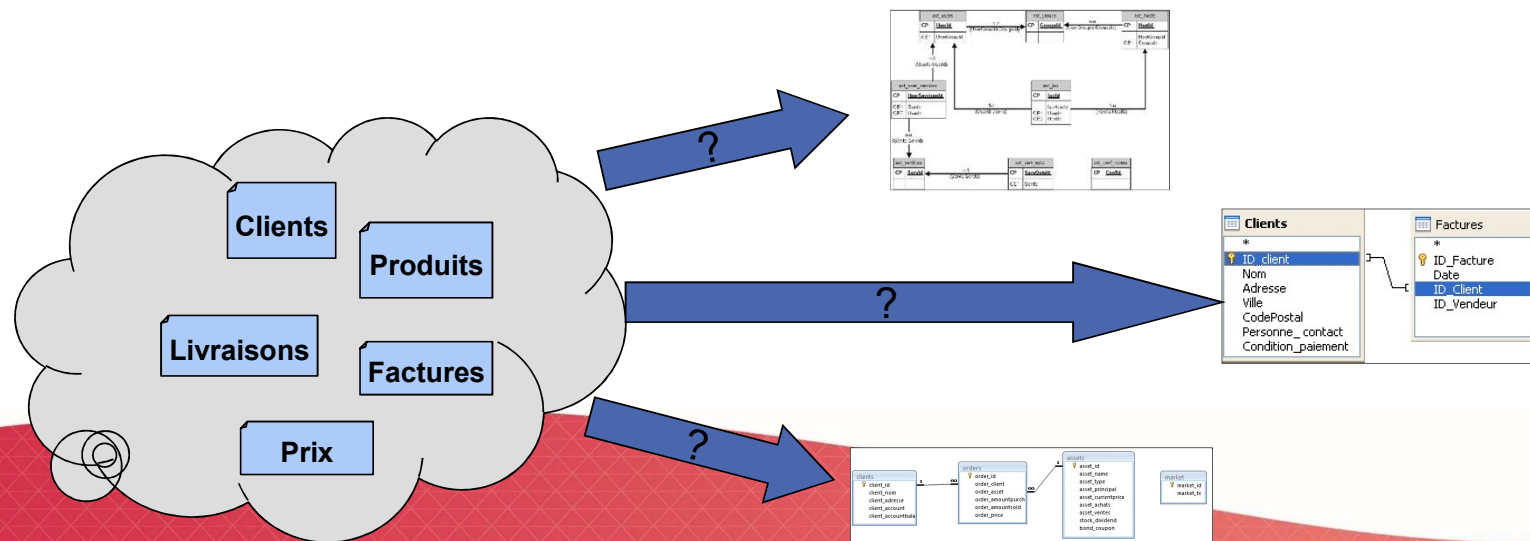
- Type **date**
  - Représente une date c'est à dire un jour, un mois et une année
- Type **timestamp**
  - Représente à la fois une date et une heure
- Type **boolean**
  - Admet les valeurs {true, false}



Il s'agit ici des types les plus communs. Les spécifications de chacun de ces types varient d'un SGBD à l'autre. De plus, chaque SGBD propose des types additionnels.

# Qualité d'un schéma

- Un même ensemble de données peut mener à divers schémas relationnels
- Il nous faut donc établir des **critères** permettant de juger de la **qualité** d'un schéma
- Il nous faut aussi des **techniques** pour **vérifier** la **qualité** d'un schéma



# Qualité d'un schéma

- Critères de qualité d'un schéma

- **Absence de redondance** des données: moins il y a de redondances et plus il est aisé de maintenir la cohérence de la BD
- **Absence d'anomalie** :
  - **Anomalies de mise à jour** : la mise à jour d'une information n'est pas répercutée dans toutes les occurrences de cette information
  - **Anomalies de suppression** : la suppression d'un tuple entraîne la perte d'une information pertinente.



# Anomalies

MEMBRE

<u>ID</u>	Nom	VoiturePreferee	Marque	Ville
1	Gaston	F40	Ferrari	LIEGE
2	Gaston	Esprit	Lotus	LIEGE
3	Michel	California	Ferrari	VERVIERS
4	Christophe	XK	Jaguar	NAMUR
5	Luc	Carrera	Porsche	ARLON

Si Gaston déménage à Visé, pensera-t-on à modifier tous les tuples ?

Si on supprime Luc, on perd l'information établissant que Carrera est un modèle de la marque Porsche.

# Les formes normales

- Des **formes normales** ont été définies pour **garantir la qualité** des schémas relationnels
- Chaque forme normale définit des **règles**
  - Une relation qui respecte ces règles appartient à cette forme
  - Il est possible de transformer une relation de manière à ce qu'elle respecte une forme normale
- Les formes normales sont **hiérarchisées**
  - Un schéma qui respecte la 3ème forme normale respecte implicitement les 1ère et 2ème formes normales
- Les formes normales peuvent s'appliquer tant sur un modèle relationnel que sur un modèle conceptuel

# 1<sup>ère</sup> forme normale (1FN)

- Une relation est en 1<sup>ère</sup> forme normale si elle possède **au moins une clé (\*)** et si **tous ses attributs sont atomiques**
  - Non décomposables
  - Mono-valués
- Cette règle est mise en œuvre en respectant les règles de transposition d'un schéma conceptuel en schéma relationnel
  - La normalisation en 1FN est obtenue en décomposant les attributs en autant d'attributs que nécessaire voire en décomposant la relation



(\*) cette condition est inhérente à la définition d'une relation

# 1<sup>ère</sup> forme normale (1FN)

- Attribut **non-décomposable**

- PERSONNE={id, nom\_prenom, rue, numéro, ville} n'est pas en 1FN
- PERSONNE={id, nom, prenom, rue, numéro, ville} est en 1FN



Le caractère non-décomposable est relatif : une date doit-elle être décomposée en jour, mois, année ?

- Attribut **mono-valué**

Etudiant
<u>Matricule</u>
Nom
Prenom
NumerosTelephone

Etudiant
<u>Matricule</u>
Nom
Prenom
NumTelephone1
NumTelephone2
NumTelephone3
NumTelephone4

Etudiant
<u>Matricule</u>
Nom
Prenom

NumTelephone
<u>NumTelephone</u>
<i>Matricule</i>



# 1<sup>ère</sup> forme normale (1FN)

- La 1FN n'est **pas suffisante** pour éviter des **incohérences** de données :
  - Soit la relation
    - COMMANDE={idCommande, numClient, numArticle, quantité, nomClient, nomArticle}
  - Les tuples
    - {1,1,2,12,"Gabor", "Bic"} et {2,1,2,10, "Gabor", "Gomme"} présentent une incohérence



L'article dont le numéro est '2' a un nom qui peut être différent suivant les commandes.

# Dépendance fonctionnelle

- Soit une relation  $R = \{A_1, A_2, \dots, A_n\}$ 
  - Soient X et Y deux sous-ensembles de R
  - On dit que Y **dépend fonctionnellement de X** ssi pour toute instance de R, la connaissance des valeurs de X permet de connaître celles de Y
  - Formellement :

$$X \rightarrow Y \text{ ssi } X(t1) = X(t2) \Rightarrow Y(t1) = Y(t2) \quad (\forall t1, t2 \in r)$$

- Une dépendance fonctionnelle  $X \rightarrow Y$  est dite « **triviale** » ssi Y est inclus dans X
- Une dépendance fonctionnelle  $X \rightarrow Y$  est dite « **élémentaire** » ssi :
  - Y est un attribut unique
  - $Y \notin X$
  - Il n'existe pas de  $X'$  inclus dans X tel que  $X' \rightarrow Y$



Cette notion sera utile pour la définition des formes normales suivantes.

# Dépendance fonctionnelle

- « On dit que **Y dépend fonctionnellement de X** ssi **pour toute instance de R, la connaissance des valeurs de X permet de connaître celles de Y** »
  - « **Y dépend fonctionnellement de X** » : peut aussi se dire « X détermine Y »
  - « **pour toute instance de R** » : pour tous les tuples (existants et à venir) de la relation R
  - « **la connaissance des valeurs de X permet de connaître celles de Y** » : si je connais la valeur de X, je peux en déduire la valeur de Y
    - **MATRICULE** → **NOM** : dans la relation ETUDIANT, si je connais la valeur de l'attribut MATRICULE, je peux déduire le NOM



Les dépendances fonctionnelles traduisent une réalité liée à la signification des attributs dans le « monde réel ».



# Dépendances fonctionnelles

- Considérons la relation ENSEIGNANT-CURSUS qui a pour schéma :
  - ENSEIGNANT-CURSUS (matricule, nom, fonction, dateengagement, idcursus, nomcursus, campus)
- On dispose des règles de gestion suivantes :
  - Un enseignant est identifié par un matricule. Il est décrit par un nom et une date d'engagement.
  - Un cursus est identifié par un numéro. Il est décrit par un nom et un campus.
  - Un enseignant peut avoir plusieurs fonctions mais uniquement au sein de cursus différents.
- *Quelles sont les dépendances fonctionnelles ?*



Que pourrait-on choisir  
comme clé primaire de  
cette relation ?

# Dépendances fonctionnelles

- Propriétés des dépendances fonctionnelles :

- Réflexivité:  $(A \subseteq B) \Rightarrow (B \rightarrow A)$
- Transitivité:  $(A \rightarrow B \wedge B \rightarrow C) \Rightarrow (A \rightarrow C)$
- Augmentation: *Soit*  $z \subset R, (A \rightarrow B) \Rightarrow (AZ \rightarrow BZ)$

Axiomes  
d'Armstrong

- Pseudo-transitivité:  $(A \rightarrow B \wedge BC \rightarrow D) \Rightarrow (AC \rightarrow D)$
- Union:  $(A \rightarrow B \wedge A \rightarrow C) \Rightarrow (A \rightarrow BC)$
- Décomposition:  $(A \rightarrow BC) \Rightarrow (A \rightarrow C) \wedge (A \rightarrow B)$
- Composition:  $(A \rightarrow B) \wedge (C \rightarrow D) \Rightarrow (AC \rightarrow BD)$



L'application de ces axiomes nous permet d'inférer l'ensemble des dépendances fonctionnelles d'une relation (fermeture transitive).

# Deuxième forme normale (2FN)

- Une relation est en deuxième forme normale ssi
  - Elle est en première forme normale
  - Aucun attribut ne **dépend fonctionnellement d'une partie de la clé**
  - **Corollaire** : une relation en 1FN dont la clé primaire est composée d'un seul attribut est, par définition, en 2FN.
- Mise en deuxième forme normale
  - Si un ensemble d'attributs dépend d'une partie de la clé ( $X \rightarrow Y$  ou  $X$  est une partie de clé)
    - On isole l'ensemble d'attributs cible et la partie de clé dépendante dans une nouvelle relation
    - On supprime l'ensemble d'attributs cible de la relation source

## Deuxième forme normale (2FN)

- Soit la relation suivante :
  - Enseignement = { num\_prof, num\_cours, nom\_prof, heure\_annee }
  - PK(num\_prof, num\_cours)
- La dépendance fonctionnelle suivante pose problème :
  - num\_prof → nom\_prof
- Passage en deuxième forme normale :
  - Enseignant = { num\_prof, nom\_prof }
  - Enseignement = { num\_prof, num\_cours, heure\_annee }

# Troisième forme normale (3FN)

- Une relation est en **troisième forme normale** ssi
  - Elle est en deuxième forme normale
  - Tout attribut n'appartenant pas à une clé ne dépend pas fonctionnellement d'un ensemble d'attributs non-clé
    - Autre expression : Pour toute DF non-triviale  $A \rightarrow B$  A est une surclé ou B fait partie d'une clé candidate
- Le respect de la 3FN permet d'éliminer les redondances dues aux dépendances transitives.

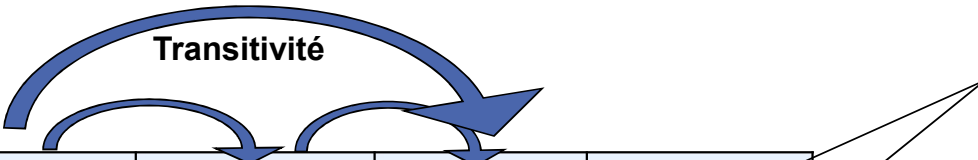
# Troisième forme normale (3FN)

Voiture={nom, modele, marque, ville} n'est pas en 3FN car :

modele → marque

nom → modele (puisque nom est la clé, il détermine tous les autres attributs de la relation)

Par transitivité : nom → marque



nom	modele	marque	ville
Gaston	F40	Ferrari	LIEGE
Marc	Esprit	Lotus	LIEGE
Roger	F40	Ferrari	VERVIERS

# Troisième forme normale (3FN)

- Pour chaque attribut non-clé déterminant d'autres attributs non-clé :
  - On isole les attributs sources et cibles dans une nouvelle relation
  - On supprime l'ensemble d'attributs cibles de la relation source

nom	modele	marque	ville
Gaston	F40	Ferrari	LIEGE
Marc	Esprit	Lotus	LIEGE
Roger	F40	Ferrari	VERVIERS

Dépendance fonctionnelle  
problématique :  
modèle → marque  
[source] → [cible]

modele	marque
F40	Ferrari
Esprit	Lotus

nom	modele	ville
Gaston	F40	LIEGE
Marc	Esprit	LIEGE
Roger	F40	VERVIERS



# Formes normales

- Un schéma en 3NF peut être qualifié de « bonne qualité » (\*)
  - peu de redondance
  - facilités d'insertion et de modification
- Avec l'expérience, les « data-modelers » créent des schémas proches de la 3NF sans devoir passer par une analyse exhaustive des relations
- Il existe d'autres formes normales (BCNF, 4FN, 5FN,...)



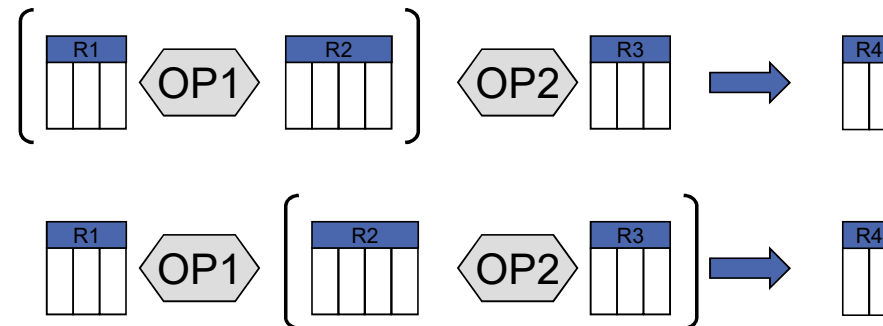
(\*) le niveau BCNF est généralement admis comme étant un niveau seuil de qualité d'un schéma. BCNF sera abordé en B2.

# Algèbre relationnelle

- Le fonctionnement interne des SGBDR repose sur 2 **notions mathématiques** fondamentales :
  - Les relations
  - L'algèbre relationnelle
- L'algèbre relationnelle offre des **moyens** pour **transformer** une ou plusieurs relations en une nouvelle relation
  - Les opérations relationnelles (fondement du langage SQL)

# Opérateurs relationnels

- L'algèbre relationnelle est donc définie par un ensemble d'**opérateurs relationnels** qui s'appliquent à une ou plusieurs relations et dont l'**évaluation** produit une nouvelle **relation**
  - Il est possible de **combiner** les opérateurs relationnels
  - Les opérateurs relationnels sont **associatifs**



# Opérateurs relationnels

- **Opérateurs unaires** : permettent d'obtenir une nouvelle relation à partir d'une autre relation
  - Sélection ( $\sigma$ ) et projection ( $\pi$ )
- **Opérateurs binaires ensemblistes** : permettent d'obtenir une nouvelle relation à partir de deux relations de même degré et dont les attributs appartiennent respectivement aux mêmes domaines
  - Union ( $\cup$ ), intersection ( $\cap$ ) et différence ( $-$ )
- **Opérateurs binaires (non-ensemblistes)** : permettent d'obtenir une nouvelle relation à partir de deux autres relations
  - Produit cartésien ( $\times$ ), jointures ( $\bowtie$ ) et division ( $\div$ )

# Sélection

- La **sélection** (aussi appelée restriction) d'une relation permet d'obtenir une nouvelle relation de **même schéma** dont les occurrences satisfont une **condition** formulée sous forme d'expression logique
- En d'autres termes, la sélection permet de **choisir** des tuples d'une relation en **fonction de conditions**
- Notation :  $\sigma_C r$

Relation			

Sélection sur la relation r  
des occurrences qui  
satisfont la condition C

# Sélection

Etudiant

Matricule	Annee_inscription	Nom	Prenom	Sexe
IG4589	2011	Dupont	Dominique	F
TC7895	2008	Fornatel	Georges	M
IG1978	2011	Gillard	Jules	M
BI4987	2009	Bruni	Evelyne	F
AU7890	2010	Prolant	Marcel	M

$\sigma_{(Sexe=M \wedge Annee\_inscription \geq 2010)}$  *Etudiant*

Matricule	Annee_inscription	Nom	Prenom	Sexe
IG1978	2011	Gillard	Jules	M
AU7890	2010	Prolant	Marcel	M

SELECT Matricule, Annee\_inscription, Nom, Prenom, Sexe  
FROM Etudiant

WHERE Sexe = 'M' AND Annee\_inscription >= 2010

# Projection

- La **projection** d'une relation permet d'obtenir une nouvelle relation dont l'ensemble des attributs est un **sous-ensemble des attributs** de la relation d'origine. Les éventuels **doublons** de cette nouvelle relation sont **supprimés**
- En d'autres termes, la projection permet de **choisir** les **attributs** d'une relation
- Notation :  $\Pi_{\{A_1, A_2, \dots, A_n\}} r$

Relation			

Projection sur la relation  $r$   
des attributs  $A_1, A_2, \dots, A_n$



# Projection

Etudiant

Matricule	Annee_inscription	Nom	Prenom	Sexe
IG4589	2011	Dupont	Dominique	F
TC7895	2008	Fornatel	Georges	M
IG1978	2011	Gillard	Jules	M
BI4987	2009	Bruni	Evelyne	F
AU7890	2010	Prolant	Marcel	M

$\Pi_{\{Matricule, Nom, Sexe\}} Etudiant$

Matricule	Nom	Sexe
IG4589	Dupont	F
TC7895	Fornatel	M
IG1978	Gillard	M
BI4987	Bruni	F
AU7890	Prolant	M

SELECT **Matricule, Nom, Sexe**  
FROM Etudiant

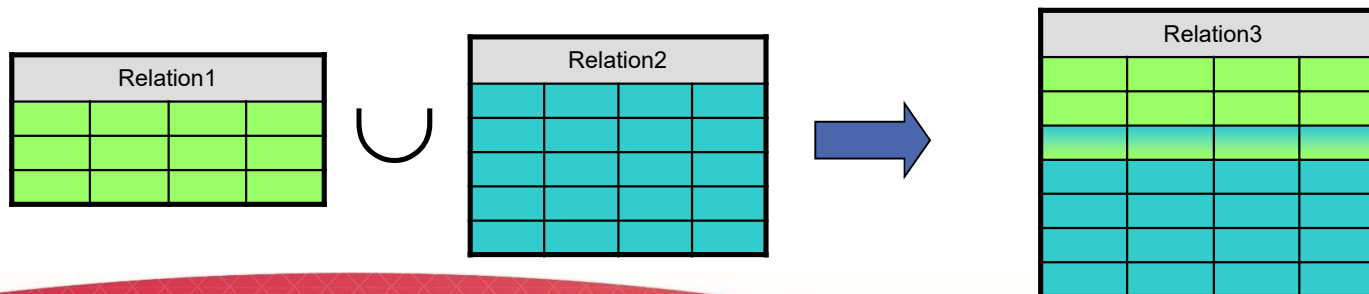
$\Pi_{\{Sexe\}} Etudiant$

Sexe
F
M

SELECT DISTINCT **Sexe**  
FROM Etudiant

# Union

- L'union de deux relations de même schéma permet d'obtenir une nouvelle relation de même schéma et constituée de tuples appartenant à une des deux relations d'origine. Les éventuels doublons de cette nouvelle relation sont supprimés
- L'union est commutative
- Notation :  $r \cup s$



# Union

Etudiant\_HEMES

Matricule	Nom	Prenom	Sexe
IG4589	Dupont	Dominique	F
TC7895	Fornatel	Georges	M
IG1978	Gillard	Jules	M
BI4987	Bruni	Evelyne	F

Etudiant\_ISELL

Matricule	Nom	Prenom	Sexe
IR4809	Arbois	Véronique	F
BI4987	Bruni	Evelyne	F
MK5895	Kalir	Ivan	M

*Etudiant\_HEMES*  $\cup$  *Etudiant\_ISELL*

Matricule	Nom	Prenom	Sexe
IG4589	Dupont	Dominique	F
TC7895	Fornatel	Georges	M
IG1978	Gillard	Jules	M
<i>BI4987</i>	<i>Bruni</i>	<i>Evelyne</i>	<i>F</i>
IR4809	Arbois	Véronique	F
MK5895	Kalir	Ivan	M

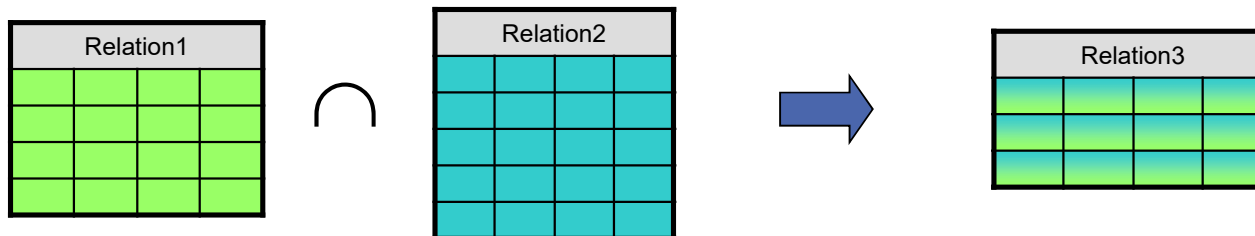
SELECT Matricule, Nom, Prenom, Sexe  
FROM Etudiant\_HEMES

**UNION**

SELECT Matricule, Nom, Prenom, Sexe  
FROM Etudiant\_ISELL

# Intersection

- L'intersection de deux relations de même schéma permet d'obtenir une nouvelle relation de même schéma et constituée de tuples appartenant aux deux relations d'origine.
- L'intersection est commutative
- Notation :  $r \cap s$



# Intersection

Etudiant\_HEMES

Matricule	Nom	Prenom	Sexe
IG4589	Dupont	Dominique	F
TC7895	Fornatel	Georges	M
IG1978	Gillard	Jules	M
BI4987	Bruni	Evelyne	F

Etudiant\_ISELL

Matricule	Nom	Prenom	Sexe
IR4809	Arbois	Véronique	F
BI4987	Bruni	Evelyne	F
MK5895	Kalir	Ivan	M

$Etudiant\_HEMES \cap Etudiant\_ISELL$

Matricule	Nom	Prenom	Sexe
BI4987	Bruni	Evelyne	F

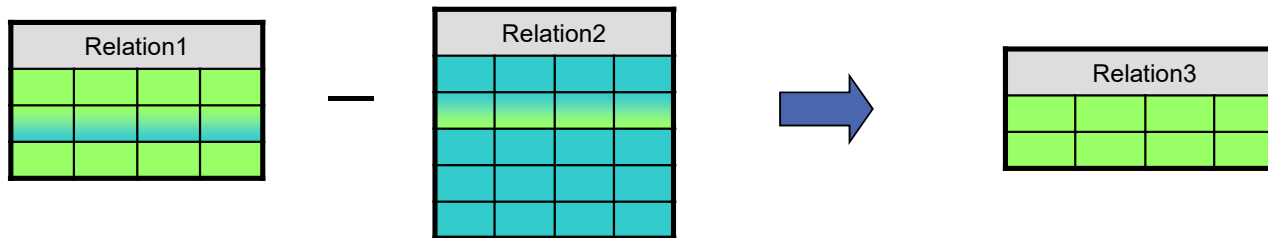
```
SELECT Matricule, Nom, Prenom, Sexe  
FROM Etudiant_HEMES
```

**INTERSECT**

```
SELECT Matricule, Nom, Prenom, Sexe  
FROM Etudiant_ISELL
```

# Différence

- La **différence** entre **deux relations** de **même schéma** permet d'obtenir une nouvelle relation de même schéma et constituée de **tuples appartenant** à la **première relation** mais pas à la deuxième
- La différence n'est pas **commutative**
- Notation :  $R - S$



# Différence

Etudiant\_HEMES

Matricule	Nom	Prenom	Sexe
IG4589	Dupont	Dominique	F
TC7895	Fornatel	Georges	M
IG1978	Gillard	Jules	M
BI4987	Bruni	Evelyne	F

Etudiant\_ISELL

Matricule	Nom	Prenom	Sexe
IR4809	Arbois	Véronique	F
BI4987	Bruni	Evelyne	F
MK5895	Kalir	Ivan	M

*Etudiant\_HEMES – Etudiant\_ISELL*

Matricule	Nom	Prenom	Sexe
IG4589	Dupont	Dominique	F
TC7895	Fornatel	Georges	M
IG1978	Gillard	Jules	M

SELECT Matricule, Nom, Prenom, Sexe  
FROM Etudiant\_HEMES

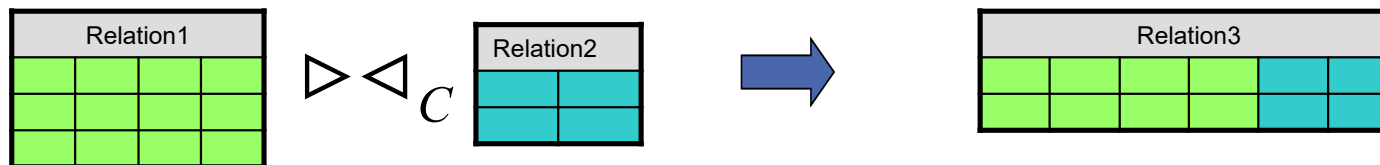
MINUS

SELECT Matricule, Nom, Prenom, Sexe  
FROM Etudiant\_ISELL



# Jointure

- La **jointure** de **deux relations** permet d'obtenir une nouvelle relation dont le schéma est constitué de **tous les attributs** des deux relations et dont les **tuples** correspondent à **l'ensemble** des **combinaisons** des tuples des deux relations qui satisfont une **condition** formulée sous forme d'expression logique
- La jointure est **commutative**
- Notation :  $r \bowtie_C s$  (équivalent à  $\sigma_c(r \times s)$ )



# Types de jointures

- **Théta-jointure** : jointure dans laquelle la condition utilise un opérateur de comparaison non-égalitaire (<,>,<=,>=,<>)
- **Equi-jointure** : jointure dans laquelle la condition utilise un opérateur de comparaison égalitaire (=)
- **Jointure naturelle** : équi-jointure portant sur des attributs de même nom qui seront fusionnés (non-dupliqués).
- **Jointure externe** : jointure pour laquelle on retourne les tuples d'une table même si la condition n'est pas remplie par rapport aux tuples de la deuxième table. Ces tuples sont complétés par des valeurs '**null**' pour les attributs non-joints.
  - Jointure externe à **gauche**
  - Jointure externe à **droite**
  - Jointure externe **complète**

# Equi-jointure

Voiture

NumSerie	Marque	Modele
1254-89-45	Peugeot	308
1547-95-13	Seat	Ibiza
2468-78-62	Peugeot	205
1548-02-95	Fiat	500

Nationalite

MarqueVoiture	Pays
Peugeot	France
Seat	Espagne
Audi	Allemagne
Fiat	Italie

*Voiture* ▷ ◁<sub>Marque=MarqueVoiture</sub> *Nationalite*

NumSerie	Marque	Modele	MarqueVoiture	Pays
1254-89-45	Peugeot	308	Peugeot	France
1547-95-13	Seat	Ibiza	Seat	Espagne
2468-78-62	Peugeot	205	Peugeot	France
1548-02-95	Fiat	500	Fiat	Italie

```
SELECT v.NumSerie, v.Marque, v.Modele,
       n.MarqueVoiture, n.Pays
FROM Voiture v
JOIN Nationalite n ON v.Marque = n.MarqueVoiture
```



La condition de jointure portera souvent sur les clés étrangères et les clés qu'elles référencent.

# Jointure naturelle

Voiture

NumSerie	Marque	Modele
1254-89-45	Peugeot	308
1547-95-13	Seat	Ibiza
2468-78-62	Peugeot	205
1548-02-95	Fiat	500

Nationalite

Marque	Pays
Peugeot	France
Seat	Espagne
Audi	Allemagne
Fiat	Italie

*Voiture*  $\triangleright \triangleleft_{\text{Marque}}$  *Nationalite*

NumSerie	Marque	Modele	Pays
1254-89-45	Peugeot	308	France
1547-95-13	Seat	Ibiza	Espagne
2468-78-62	Peugeot	205	France
1548-02-95	Fiat	500	Italie

# Jointure externe à gauche

Voiture

NumSerie	Marque	Modele
1254-89-45	Peugeot	308
1547-95-13	Seat	Ibiza
2468-78-62	Peugeot	205
3658-81-02	BMW	X5
1548-02-95	Fiat	500

Nationalite

MarqueVoiture	Pays
Peugeot	France
Seat	Espagne
Audi	Allemagne
Fiat	Italie

*Voiture* ▷ ◁ *Nationalite* — *Marque = MarqueVoiture*

NumSerie	Marque	Modele	MarqueVoiture	Pays
1254-89-45	Peugeot	308	Peugeot	France
1547-95-13	Seat	Ibiza	Seat	Espagne
2468-78-62	Peugeot	205	Peugeot	France
3658-81-02	BMW	X5	NULL	NULL
1548-02-95	Fiat	500	Fiat	Italie

# Jointure externe à droite

Voiture

NumSerie	Marque	Modele
1254-89-45	Peugeot	308
1547-95-13	Seat	Ibiza
2468-78-62	Peugeot	205
3658-81-02	BMW	X5
1548-02-95	Fiat	500

Nationalite

MarqueVoiture	Pays
Peugeot	France
Seat	Espagne
Audi	Allemagne
Fiat	Italie

*Voiture* —  $\triangleright \triangleleft_{\text{Marque} = \text{MarqueVoiture}}$  *Nationalite*

NumSerie	Marque	Modele	MarqueVoiture	Pays
1254-89-45	Peugeot	308	Peugeot	France
1547-95-13	Seat	Ibiza	Seat	Espagne
2468-78-62	Peugeot	205	Peugeot	France
<b>NULL</b>	<b>NULL</b>	<b>NULL</b>	Audi	Allemagne
1548-02-95	Fiat	500	Fiat	Italie

# Jointure externe complète

Voiture

NumSerie	Marque	Modele
1254-89-45	Peugeot	308
1547-95-13	Seat	Ibiza
2468-78-62	Peugeot	205
3658-81-02	BMW	X5
1548-02-95	Fiat	500

Nationalite

MarqueVoiture	Pays
Peugeot	France
Seat	Espagne
Audi	Allemagne
Fiat	Italie

*Voiture* – ▷◁ – *Nationalit e*  
Marque = MarqueVoit ure

NumSerie	Marque	Modele	MarqueVoiture	Pays
1254-89-45	Peugeot	308	Peugeot	France
1547-95-13	Seat	Ibiza	Seat	Espagne
2468-78-62	Peugeot	205	Peugeot	France
3658-81-02	BMW	X5	<b>NULL</b>	<b>NULL</b>
<b>NULL</b>	<b>NULL</b>	<b>NULL</b>	Audi	Allemagne
1548-02-95	Fiat	500	Fiat	Italie