

# DIU-EIL BLOC 4

## MODÈLE RELATIONNEL ET BASES DE DONNÉES : NORMALISATION ET CONCEPTION

Romuald THION

<https://forge.univ-lyon1.fr/diu-eil/bloc4>

25 mai 2020

# Plan

- 1 Problème et motivation : les anomalies
- 2 La théorie de la normalisation
- 3 La normalisation
- 4 Les contraintes d'intégrité en SQL

1 Problème et motivation : les anomalies

2 La théorie de la normalisation

3 La normalisation

4 Les contraintes d'intégrité en SQL

# Objectifs

## Modéliser

Modéliser consiste à définir un monde abstrait qui coïncide avec les manifestations apparentes du monde réel.

- il s'agit donc de déterminer l'ensemble des attributs, des relations et des contraintes qui constitueront le modèle.

## Nous allons voir :

- Quelles sont les propriétés attendues d'une **bonne** modélisation ;
- Comment les obtenir.

modéliser  $\Rightarrow$  représenter le réel d'une façon simplifiée

On va parler des contraintes

Propriétés attendues d'une bonne modélisation.

# Position dans le programme de Term NSI

Problème et motivation : les anomalies		
Positionnement dans le programme		
Contenus	Capacités attendues	Commentaires
Modèle relationnel : relation, attribut, domaine, clé primaire, clé étrangère, schéma relationnel.	Identifier les concepts définissant le modèle relationnel.	Ces concepts permettent d'exprimer les contraintes d'intégrité (domaine, relation et référence).
Base de données relationnelles.	Savoir distinguer la structure d'une base de données de son contenu.  Repérer des anomalies dans le schéma d'une base de données.	La structure est un ensemble de schémas relationnels qui respecte les contraintes du modèle relationnel.  Les anomalies peuvent être des redondances de données ou des anomalies d'insertion, de suppression, de mise à jour. On priviliege la manipulation de données nombreuses et réalistes.
Système de gestion de bases de données relationnelles.	Identifier les services rendus par un système de gestion de bases de données relationnelles : persistance des données, gestion des accès concurrents, efficacité de traitement des requêtes, sécurisation des accès.	Il s'agit de comprendre le rôle et les enjeux des différents services sans détailler le fonctionnement.

# Positionnement dans le programme

Contenus	Capacités attendues	Commentaires
Modèle relationnel : relation, attribut, domaine, clef primaire, clef étrangère, schéma relationnel.	Identifier les concepts définissant le modèle relationnel.	Ces concepts permettent d'exprimer les contraintes d'intégrité (domaine, relation et référence).
Base de données relationnelle.	<p>Savoir distinguer la structure d'une base de données de son contenu.</p> <p>Repérer des anomalies dans le schéma d'une base de données.</p>	<p>La structure est un ensemble de schémas relationnels qui respecte les contraintes du modèle relationnel.</p> <p>Les anomalies peuvent être des redondances de données ou des anomalies d'insertion, de suppression, de mise à jour.</p> <p>On priviliege la manipulation de données nombreuses et réalistes.</p>
Système de gestion de bases de données relationnelles.	Identifier les services rendus par un système de gestion de bases de données relationnelles : persistance des données, gestion des accès concurrents, efficacité de traitement des requêtes, sécurisation des accès.	Il s'agit de comprendre le rôle et les enjeux des différents services sans en détailler le fonctionnement.

# Exemple

Soit  $\mathcal{U} = \{\text{NumEt}, \text{NomEt}, \text{Adresse}, \text{NumUE}, \text{Titre}, \text{Note}\}$  l'ensemble d'attributs (appelé *univers*) décrivant des étudiants et des cours.  
Soient les deux schémas de BD suivants :

- $BD_1 = \{\text{Donnees}\}$  avec  $\text{schema}(\text{Donnees}) = \mathcal{U}$   
*une seule relation qui contient tout, comme dans un tableau*
- $BD_2 = \{\text{Etudiant}, \text{UE}, \text{Inscrit}\}$  avec
  - $\text{schema}(\text{Etudiant}) = \{\text{NumEt}, \text{NomEt}, \text{Adresse}\}$
  - $\text{schema}(\text{UE}) = \{\text{NumUE}, \text{Titre}\}$
  - $\text{schema}(\text{Inscrit}) = \{\text{NumEt}, \text{NumUE}, \text{Note}\}$

Est-ce que ce serait mieux,  
d'avoir une seule relation  
plutôt que de séparer ?  
Pourquoi séparer ?

### Exemple

Soit  $\mathcal{U} = \{\text{NumEt}, \text{NomEt}, \text{Adresse}, \text{NumUE}, \text{Titre}, \text{Note}\}$  l'ensemble d'attributs (appelé univers) décrivant des étudiants et des cours.  
Soient les deux schémas de BD suivants :

- $BD_1 = \{\text{Donnees}\}$  avec  $\text{schema}(\text{Donnees}) = \mathcal{U}$   
une seule relation qui contient tout, comme dans un tableau
- $BD_2 = \{\text{Etudiant}, \text{UE}, \text{Inscrit}\}$  avec
  - $\text{schema}(\text{Etudiant}) = \{\text{NumEt}, \text{NomEt}, \text{Adresse}\}$
  - $\text{schema}(\text{Cours}) = \{\text{NumUE}, \text{Titre}\}$
  - $\text{schema}(\text{Inscrit}) = \{\text{NumEt}, \text{NumUE}, \text{Note}\}$

Prenons l'exemple d'une base de données avec une seule table qui ressemble à un tableau:

- Premier problème qu'on peut avoir : **l'anomalie de modification**  
→ toute modification doit être répétée éventuellement sur plusieurs lignes à cause de la redondance des données

⇒ modification beaucoup moins atomiques

Anomalie de modification

NumEt	NomEt	Adresse	NUMUE	Titre	Note
124	Jean	Paris	F234	Philo I	A
456	Emma	Lyon	F234	Philo I	B
789	Paul	Marseille	M321	Analyse I	C
124	Jean	Paris	M321	Analyse I	A
789	Paul	Marseille	CS24	BD I	B

Anomalie de modification

Une modification sur une ligne peut nécessiter des modifications sur d'autres lignes.

⇒ Si on ne le fait pas, on va introduire des incohérences

# Exemple

NumEt	NomEt	Adresse	NumUE	Titre	Note
124	Jean	Paris	F234	Philo I	A
456	Emma	Lyon	F234	Philo I	B
789	Paul	Marseille	M321	Analyse I	C
124	Jean	Paris	M321	Analyse I	A
789	Paul	Marseille	CS24	BD I	B

TABLE – La relation universelle de tous les attributs de l'univers  $\mathcal{U}$

Comment évaluer ces deux schémas ? Lequel est meilleur ?  
Pourquoi ? Selon quels critères ?

## Anomalie de *modification*

NumEt	NomEt	Adresse	NumUE	Titre	Note
124	Jean	Paris	F234	Philo I	A
456	Emma	Lyon	F234	Philo I	B
789	Paul	Marseille	M321	Analyse I	C
124	Jean	Paris	M321	Analyse I	A
789	Paul	Marseille	CS24	BD I	B

## Anomalie de *modification*

*Une modification sur une ligne peut nécessiter des modifications sur d'autres lignes.*

## Anomalie de suppression

- Par exemple, si on supprime la dernière ligne parce que Paul n'est plus inscrit à l'VE CS24, on perd en même temps le lien entre le numéro de cette VE et son titre

## Anomalie de suppression

NumEt	NomEt	Adresse	NumUE	Titre	Note
124	Jean	Paris	F234	Philo I	A
456	Emma	Lyon	F234	Philo I	B
789	Paul	Marseille	M321	Analyse I	C
124	Jean	Paris	M321	Analyse I	A
789	Paul	Marseille	CS24	BD I	B

## Anomalie de suppression

Certaines informations dépendent de l'existence d'autres informations.

l'inscription de

En supprimant Paul, on supprime  
une autre information  
Certaines informations  
dépendent de l'existence  
d'autres informations

# Anomalie de suppression

NumEt	NomEt	Adresse	NumUE	Titre	Note
124	Jean	Paris	F234	Philo I	A
456	Emma	Lyon	F234	Philo I	B
789	Paul	Marseille	M321	Analyse I	C
124	Jean	Paris	M321	Analyse I	A
789	Paul	Marseille	CS24	BD I	B

## Anomalie de suppression

Certaines informations dépendent de l'existence d'autres informations.

Dual de l'anomalie de suppression : L'anomalie d'insertion

Si on a juste un nouvel élément  
(ou une nouvelle UE dans inscrit)  
on va avoir un pb

Problème et motivation : les anomalies

### Anomalie d'insertion

NumEt	NomEt	Adresse	NumUE	Titre	Note
124	Jean	Paris	F234	Philo I	A
456	Emma	Lyon	F234	Philo I	B
789	Paul	Marseille	M321	Analyse I	C
124	Jean	Paris	M321	Analyse I	A
789	Paul	Marseille	CS24	BD I	B
145	Evariste	Aubenas	???	???	???

Anomalie d'insertion

La possibilité d'enregistrer un tuple implique la connaissance de toutes les informations qui lui sont liées : problème de valeurs manquantes.



R. THION DIU-BI Bloc 4 - BD 10

# Anomalie d'insertion

NumEt	NomEt	Adresse	NumUE	Titre	Note
124	Jean	Paris	F234	Philo I	A
456	Emma	Lyon	F234	Philo I	B
789	Paul	Marseille	M321	Analyse I	C
124	Jean	Paris	M321	Analyse I	A
789	Paul	Marseille	CS24	BD I	B
145	Evariste	Aubenas	???	???	???

## Anomalie d'insertion

La possibilité d'enregistrer un tuple implique la connaissance de toutes les informations qui lui sont liées : problème de valeurs manquantes.

# Comment formaliser tout ?

- faire apparaître les dépendances fonctionnelles
- théorie de la normalisation.
- quelques liens vers des ressources de licences sur le GitLab

# Comment formaliser tout ça ?

Le moyen qui permet d'éviter ces problèmes est l'étude des dépendances et de la normalisation.

1 Problème et motivation : les anomalies

2 La théorie de la normalisation

- Les dépendances fonctionnelles
- Les formes normales

3 La normalisation

4 Les contraintes d'intégrité en SQL

1 Problème et motivation : les anomalies

2 La théorie de la normalisation

- Les dépendances fonctionnelles
- Les formes normales

3 La normalisation

4 Les contraintes d'intégrité en SQL

## Définition

- La forme la plus fréquemment rencontrée de dépendances.
- Formalisent la notion de clef (identifiant) d'une relation.
- Permettent de définir les « bon » schémas (sans redondance)

### Syntaxe des dépendances fonctionnelles

Une *Dépendance Fonctionnelle (DF)* sur un schéma de relation  $R \subseteq U$  est une expression de la forme

DF ?       $R : X \rightarrow Y$ , avec  $X, Y \subseteq R$

- Une DF  $X \rightarrow Y$  est dite **triviale** si  $Y \subseteq X$
- Une DF **standard** si  $X \neq \emptyset$ .



### Sémantique des dépendances fonctionnelles

Soit  $r$  une instance de relation sur  $R$ . Une DF  $R : X \rightarrow Y$  est satisfaite dans  $r$ , noté  $r \models X \rightarrow Y$ ,ssi

$$\forall t_1, t_2 \in r \quad t_1[X] = t_2[X] \Rightarrow t_1[Y] = t_2[Y]$$

On dit aussi que  $X$  détermine (fonctionnellement)  $Y$  dans  $r$ .

Une dépendance fonctionnelle est satisfait si dès qu'on a 2 tuples, si ils ont la même projection

sur  $X$  alors ils ont la  
même projection sur  $Y$ .  
(notion d'implication logique)

## Les dépendances fonctionnelles

Soient  $\mathcal{U}$  un ensemble d'attributs et  $\mathcal{D}$  un domaine et  $R \subseteq \mathcal{U}$  :

- un tuple  $t$  de  $R$  est une fonction  $R \rightarrow \mathcal{D}$
- une instance  $r$  de  $R$  est un ensemble fini de tuples
- la projection de  $t$  sur  $X \subseteq R$  notée  ${}^{\hat{a}} t[X]$  est la restriction de  $t$  à  $X$ 
  - a. On écrit plutôt  $t|_X$  en mathématiques usuelles

# Définition

- La forme la plus fréquemment rencontrée de dépendances.
- Formalisent la notion de **clef** (identifiant) d'une relation.
- Permettent de définir les « bon » schémas (sans redondance)

## Syntaxe des dépendances fonctionnelles

Une *Dépendance Fonctionnelle (DF)* sur un schéma de relation  $R \subseteq \mathcal{U}$  est une expression de la forme

$$R : X \rightarrow Y, \text{ avec } X, Y \subseteq R$$

- Une DF  $X \rightarrow Y$  est dite **triviale** si  $Y \subseteq X$
- Une DF **standard** si  $X \neq \emptyset$ .

# Les dépendances fonctionnelles

Soient  $\mathcal{U}$  un ensemble d'attributs et  $\mathcal{D}$  un domaine et  $R \subseteq \mathcal{U}$  :

- un tuple  $t$  de  $R$  est une fonction  $R \rightarrow \mathcal{D}$
  - une instance  $r$  de  $R$  est un ensemble fini de tuples
  - la projection de  $t$  sur  $X \subseteq R$  notée <sup>a</sup>  $t[X]$  est la restriction de  $t$  à  $X$
- 
- a. On écrit plutôt  $t|_X$  en mathématiques usuelles

## Sémantique des dépendances fonctionnelles

Soit  $r$  une instance de relation sur  $R$ . Une DF  $R : X \rightarrow Y$  est satisfaite dans  $r$ , noté  $r \models X \rightarrow Y$ ,ssi

$$\forall t_1, t_2 \in r. t_1[X] = t_2[X] \Rightarrow t_1[Y] = t_2[Y]$$

On dit aussi que  $X$  détermine (fonctionnellement)  $Y$  dans  $r$ .

- Illustration d'une dépendance fonctionnelle.

$\pi \models \text{NurEt} \rightarrow \text{NonEt}$

notation

les colonnes NurEt, NonEt forment le graphe de cette fonction

## Les dépendances fonctionnelles

Exemple

r	NumEt	NomEt	Adresse	NumUE	Titre	Note
124	Jean	Paris	F234	Philo I	A	
456	Emma	Lyon	F234	Philo I	B	
789	Paul	Marseille	M321	Analyse I	C	
124	Jean	Paris	M321	Analyse I	A	
789	Paul	Marseille	CS24	BD I	B	

- $r \models \underline{\text{NumEt}} \rightarrow \underline{\text{NomEt}}$  et  $r \models \text{NumEt}, \text{NumUE} \rightarrow \text{Note}$
- $r \models \text{Adresse} \rightarrow \text{NumEt}$  ( $\dagger$ )
- $r \not\models \text{NumEt} \rightarrow \text{NumUE}$  et  $r \not\models \text{NumUE} \rightarrow \text{Note}$

On a une dépendance fonctionnelle entre les attributs

- si la restriction de la relation R aux attributs X et Y est le graphe d'une fonction en sens mathématique
- type des termes

Dépendances fonctionnelles  
qu'on peut déduire logique-  
ment

La théorie de la normalisation | Les dépendances fonctionnelles

### Les dépendances fonctionnelles

Exemple

r	NumEt	NomEt	Adresse	NumUE	Titre	Note
124	Jean	Paris	F234	Philo I	A	
456	Emma	Lyon	F234	Philo I	B	
789	Paul	Marseille	M321	Analyse I	C	
124	Jean	Paris	M321	Analyse I	A	
789	Paul	Marseille	CS24	BD I	B	

•  $r \vdash \text{NumEt} \rightarrow \text{NomEt}$  et  $r \vdash \text{NumEt}^h, \text{NumUE} \rightarrow \text{Note}$

•  $r \vdash \text{Adresse} \rightarrow \text{NumEt}$  (?)

•  $r \not\vdash \text{NumEt} \rightarrow \text{NumUE}$  et  $r \not\vdash \text{NumUE} \rightarrow \text{Note}$

Une dépendance fonctionnelle qui existe mais qui est liée aux données? Si on a :

La théorie de la normalisation Les dépendances fonctionnelles

### Les dépendances fonctionnelles

Exemple

r	NumEt	NomEt	Adresse	NumUE	Titre	Note
124	Jean	Paris	F234	Philo I	A	
456	Emma	Lyon	F234	Philo I	B	
789	Paul	Marseille	M321	Analyse I	C	
124	Jean	Paris	M321	Analyse I	A	
789	Paul	Marseille	CS24	BD I	B	

- $r \vdash \text{NumEt} \rightarrow \text{NomEt}$  et  $r \vdash \text{NumEt}, \text{NumUE} \rightarrow \text{Note}$
- $r \vdash \text{Adresse} \rightarrow \text{NumEt}$  (?)
- $r \not\vdash \text{NumEt} \rightarrow \text{NumUE}$  et  $r \not\vdash \text{NumUE} \rightarrow \text{Note}$

$r \vdash \text{Adresse} \rightarrow \text{NumEt}$   
cela veut dire  
qu'on a au plus  
un étudiant  
par ville

Contraintes imposées par une dépendance fonctionnelle:

$$r \models X \rightarrow Y$$

plus  $X$  est réduit plus la contrainte est forte (par exemple si  $X = \emptyset$   $Y$  doit être constante)

Plus  $X$  est large / grand  
, moins la contrainte sol  
forte .

Cas où on n'a pas de dépendances fonctionnelles  
 $r \models \text{NumEt} \rightarrow \text{NumUE}$

Un étudiant peut être inscrit dans plusieurs VE

La théorie de la normalisation | Les dépendances fonctionnelles

### Les dépendances fonctionnelles

Exemple

$r$	NumEt	NomEt	Adresse	NumUE	Titre	Note
	124	Jean	Paris	F234	Philo I	A
	456	Emma	Lyon	F234	Philo I	B
	789	Paul	Marseille	M321	Analyse I	C
	124	Jean	Paris	M321	Analyse I	A
	789	Paul	Marseille	CS24	BD I	B

•  $r \models \text{NumEt} \rightarrow \text{NomEt}$  et  $r \models \text{NumEt}, \text{NumUE} \rightarrow \text{Note}$

•  $r \models \text{Adresse} \rightarrow \text{NumEt}$  ( $\dagger$ )

•  $r \not\models \text{NumEt} \rightarrow \text{NumUE}$  et  $r \not\models \text{NumUE} \rightarrow \text{Note}$

# Les dépendances fonctionnelles

## Exemple

<i>r</i>	NumEt	NomEt	Adresse	NumUE	Titre	Note
124	Jean	Paris	F234	Philo I	A	
456	Emma	Lyon	F234	Philo I	B	
789	Paul	Marseille	M321	Analyse I	C	
124	Jean	Paris	M321	Analyse I	A	
789	Paul	Marseille	CS24	BD I	B	

- $r \models \text{NumEt} \rightarrow \text{NomEt}$  et  $r \models \text{NumEt}, \text{NumUE} \rightarrow \text{Note}$
- $r \models \text{Adresse} \rightarrow \text{NumEt}$  ( $\dagger$ )
- $r \not\models \text{NumEt} \rightarrow \text{NumUE}$  et  $r \not\models \text{NumUE} \rightarrow \text{Note}$

Une clé est un ensemble d'attributs  $X$  qui détermine toute la relation  $R$  par dépendance fonctionnelle.

$\pi$ :

$$\pi : X \rightarrow R$$

# Les dépendances fonctionnelles

## Clef

Une clef peut-être définie de deux manières équivalentes :

- Une clef est un ensemble d'attributs  $X$  qui ne prend jamais deux fois la même valeur dans  $r$ .  
- Une clef est un ensemble d'attributs qui détermine tout  $R$ , c'est-à-dire tel que  $r : X \rightarrow R$

## Clef primaire (*primary key*)

Une clef primaire est simplement une clef parmi les autres (appelées *clefs candidates*), choisie par le concepteur pour sa simplicité ou son aspect naturel.

1 Problème et motivation : les anomalies

2 La théorie de la normalisation

- Les dépendances fonctionnelles
- Les formes normales

3 La normalisation

4 Les contraintes d'intégrité en SQL

Pour normaliser la relation on va s'appuyer sur les dépendances fonctionnelles

Il faut que les dépendances fonctionnelles qui restent dans les tables soient des clefs.

## Les formes normales

La solution aux problèmes des anomalies consiste à **normaliser la relation** en la décomposant. Cette décomposition s'appuie sur les dépendances fonctionnelles qui existent entre les attributs.

Les formes normales permettent de spécifier formellement la notion *intuitive* de « bon schéma »

L'idée générale est de n'avoir **que les clés** à vérifier et d'éliminer au maximum des DF qui ne définissent pas des clés.

Plus on va vers le noyau, plus il y a de contraintes fortes.  
Et moins il y a d'anomalies.

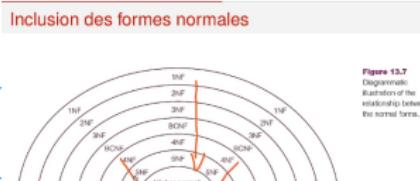
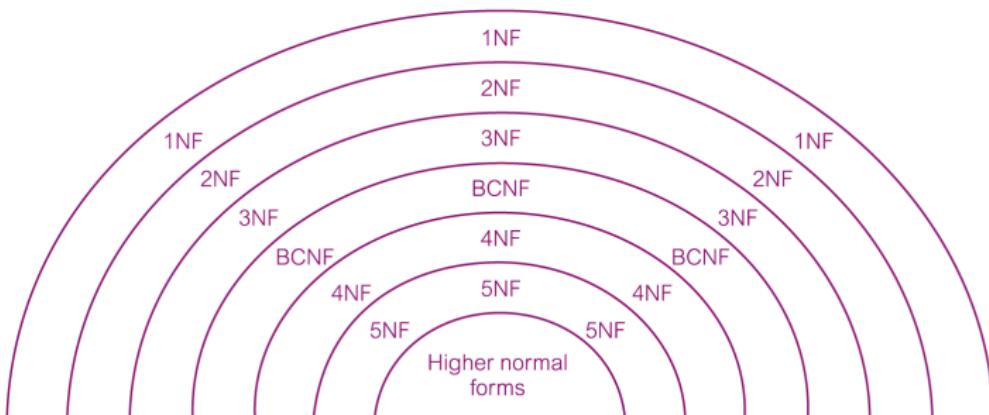


FIGURE – Pour les DFs, plusieurs Formes Normales (FN) de plus en plus restrictives (1FN, 2FN, 3FN, FN de Boyce-Codd – FNBC). Au delà, d'autres types de dépendances sont nécessaires.

Comme s'intègre  
- sera qu'à  
la forme normale  
de Boyce-Codd  
FNBC

# Inclusion des formes normales



**Figure 13.7**

Diagrammatic illustration of the relationship between the normal forms.

**FIGURE –** Pour les DFs, plusieurs Formes Normales (FN) de plus en plus restrictives (1FN, 2FN, 3FN, FN de Boyce-Codd – FNBC). Au delà, d'autres types de dépendances sont nécessaires.

Voir Laurent Andlauer "Bases de données  
de la modélisation au SQL" ns. 84 et 85

- 1FN : tout attribut contient une valeur atomique : pas multiple, pas composée
- 2FN :
  - en 1FN
  - toutes les dépendances fonctionnelles entre la clé et les autres attributs sont élémentaires.

Rque : Une relation peut être en 2FN par rapport à l'une de ses clés et non

par rapport à une autre.

- 3FN :
  - en 2 FN
    - tout attribut n'appartenant pas à une clé n'est pas en dépendance fonctionnelle directe (sans transitivité) avec un ensemble d'attributs non clés

Règle: Une relation peut être en 3 FN pour une clé mais pas pour les autres

FNB C. les seules dépendances fonctionnelles élémentaires sont celles dans lesquelles une clé détermine un attribut non clé.

Rqee: Si une relation est en BCNF pour une clé alors elle l'est pour toutes ses clés

## Les formes normales

Soit  $R$  un schéma de relation et  $F$  un ensemble de DF définies sur  $R$ .  
Un schéma de BD est en  $n$ FN si tous ses schémas de relations le sont.

Des contraintes de plus en plus restrictives

**1FN** toutes les valeurs des attributs sont **atomiques**

**2FN**  $R$  est en 1FN et aucun attribut non clef ne dépend **partiellement** d'une clef candidate.

E.g.,  $\{AB \rightarrow C, B \rightarrow C\}$  n'est pas en 2FN.

**3FN**  $R$  est en 2FN et toutes les DFs sont directes : tout attribut non clef dépend **directement** d'une clé (sans transitivité).

E.g.,  $\{A \rightarrow B, B \rightarrow C\}$  est en 2FN mais pas en 3FN.

**FNBC**  $R$  est en FN de Boyce-Codd ssi pour chaque DF  $X \rightarrow A$  de  $F$ ,  $X$  contient une clef de  $R$ .

E.g.,  $\{AB \rightarrow C, C \rightarrow B\}$  est en 3FN mais pas en FNBC.

- clef
- $\{ \overline{AB} \rightarrow C, B \rightarrow C \}$  n'est pas en 2 FN

car la dépendance fonctionnelle

$AB \rightarrow C$  n'est pas élémentaire

- $\{ A \rightarrow B, B \rightarrow C \}$  est en 2 FN

clef

mais pas en 3 FN car l'attribut C n'appartient pas à la clef

est en

dépendance directe avec B  
qui n'appartient pas à la clé

bf

- $\{\overline{AB} \rightarrow C, C \rightarrow B\}$  est en F<sub>2N</sub>  
et en F<sub>3N</sub> mais pas en F<sub>NBC</sub>  
en l'attribut B qui appartient  
à la clé est en dépendance directe  
avec C qui n'appartient pas à  
la clé : une partie de la clé (B)  
est déterminée par un attribut non clé (C)

## La FNBC : une forme idéale

La FNBC est, pour les DF, la forme idéale d'un schéma de BD  
« La clef, toute la clef et rien que la clef. »

Formalisation de la notion de redondance avec les DF

Ils existent  $X \rightarrow A \in F$  et  $t_i \neq t_j \in r$  t.q.  $t_i[XA] = t_j[XA]$

Les trois propriétés suivantes sont équivalentes

- $R$  est en FNBC par rapport à  $F$
- $R$  n'a pas de problème de redondances par rapport à  $F$
- ( $R$  n'a pas de problème de mise-à-jour par rapport à  $F$ )



## Reprise de l'exemple

	NumEt	NomEt	Adresse	NumUE	Titre	Note
$t_1$	124	Jean	Paris	F234	Philo I	A
$t_2$	456	Emma	Lyon	F234	Philo I	B
$t_3$	789	Paul	Marseille	M321	Analyse I	C
$t_4$	124	Jean	Paris	M321	Analyse I	A
$t_5$	789	Paul	Marseille	CS24	BD I	B

Vers un bon schéma

Ici, beaucoup de redondances, car  $\text{NumEt} \rightarrow \text{NomEt}, \text{Adresse}$  et  
 $\text{NumUE} \rightarrow \text{Titre}$  et  $\text{NumEt}, \text{NumUE} \rightarrow \text{Note}$ .

Mais on voit « un bon schéma » commencer à apparaître !

# Les formes normales

Soit  $R$  un schéma de relation et  $F$  un ensemble de DF définies sur  $R$ .  
Un schéma de BD est en  $n$ FN si tous ses schémas de relations le sont.

## Des contraintes de plus en plus restrictives

**1FN** toutes les valeurs des attributs sont **atomiques**

**2FN**  $R$  est en 1FN et aucun attribut non clef ne dépend **partiellement** d'une clef candidate.

*E.g.,  $\{AB \rightarrow C, B \rightarrow C\}$  n'est pas en 2FN.*

**3FN**  $R$  est en 2FN et toutes les DFs sont directes : tout attribut non clef dépend **directement** d'une clé (sans transitivité).

*E.g.,  $\{A \rightarrow B, B \rightarrow C\}$  est en 2FN mais pas en 3FN.*

**FNBC**  $R$  est en FN de Boyce-Codd ssi pour chaque DF  $X \rightarrow A$  de  $F$ ,  $X$  contient une clef de  $R$ .

*E.g.,  $\{AB \rightarrow C, C \rightarrow B\}$  est en 3FN mais pas en FNBC.*

## La FNBC : une forme idéale

La FNBC est, *pour les DF*, la forme idéale d'un schéma de BD  
« *La clef, toute la clef et rien que la clef.* »

### Formalisation de la notion de redondance avec les DF

Ils existent  $X \rightarrow A \in F$  et  $t_i \neq t_j \in r$  t.q.  $t_i[XA] = t_j[XA]$

Les trois propriétés suivantes sont équivalentes

- $R$  est en FNBC par rapport à  $F$
- $R$  n'a pas de problème de redondances par rapport à  $F$
- ( $R$  n'a pas de problème de mise-à-jour par rapport à  $F$ )

# Reprise de l'exemple

	NumEt	NomEt	Adresse	NumUE	Titre	Note
$t_1$	124	Jean	Paris	F234	Philo I	A
$t_2$	456	Emma	Lyon	F234	Philo I	B
$t_3$	789	Paul	Marseille	M321	Analyse I	C
$t_4$	124	Jean	Paris	M321	Analyse I	A
$t_5$	789	Paul	Marseille	CS24	BD I	B

## Vers un bon schéma

Ici, beaucoup de redondances, car  $\text{NumEt} \rightarrow \text{NomEt}, \text{Adresse}$  et  
 $\text{NumUE} \rightarrow \text{Titre}$  et  $\text{NumEt}, \text{NumUE} \rightarrow \text{Note}$ .

Mais on voit « un bon schéma » commencer à apparaître !

1 Problème et motivation : les anomalies

2 La théorie de la normalisation

3 La normalisation

- Les approches calculatoires
- Les diagramme Entités-Associations

4 Les contraintes d'intégrité en SQL

# Normaliser

L'activité qui consiste, étant donné un ensemble de DF<sup>1</sup>, à avoir un schéma *en bonne forme normale*.

- Soit de façon **calculatoire**, avec l'aide des DFs et d'algorithmes :
  - *Par synthèse* : on génère les schémas de relation à partir des DFs
  - *Par décomposition* : on raffine successivement  $\mathcal{U}$
- Soit par **construction**, avec les schéma Entités-Associations (E/A)

## Exemple filé de la base étudiant

On obtient (par synthèse ou décomposition) :

- $R_0(\text{NumEt}, \text{NomEt}, \text{Adresse})$ , i.e., Etudiant !
- $R_1(\text{NumUE}, \text{Titre})$ , i.e., UE !
- $R_2(\text{NumEt}, \text{NumUE}, \text{Note})$ , i.e., Inscrit !

1. On définit les DFs *a priori*, à partir de la sémantique des données, on ne part pas d'une instance qui pourrait vérifier « par hasard »  $r \models \text{Adresse} \rightarrow \text{NumEt}$  !

## Quand ne *pas* normaliser ?

La normalisation n'est *pas* une obligation on peut vouloir s'en passer :

- Pour retrouver « toutes » les données (originales), il faut calculer des jointures, qui peuvent être **coûteuses** :
  - elle sont généralement nombreuses car la décomposition est maximale,
  - leur calcul n'est pas toujours performant, en particulier si les index ne sont pas adaptés.
- La normalisation peut être difficile et donc coûteuse.
- On en a pas nécessairement besoin quand la base n'a pas une très grande durée de vie ou s'il n'y a jamais de modifications.

1 Problème et motivation : les anomalies

2 La théorie de la normalisation

3 La normalisation

- Les approches calculatoires
- Les diagramme Entités-Associations

4 Les contraintes d'intégrité en SQL

Sans perte de dépendances :

Si on a Num Et  $\rightarrow$  Nom

on ne veut pas séparer NumEt et Nom car NumEt détermine le nom.

$\hookrightarrow$  on y arrive toujours en 3FN

mais pas toujours en forme normale de Boyce Codd

contre-exemple à une FNBC  
et génération mise en FNAC

La normalisation

Les approches calculatoires

## Algorithme de décomposition

### Algorithme de décomposition en FNBC

Soit  $S = \{R\}$ . Tant qu'il existe dans  $R_i \in S$  qui n'est pas en FNBC :

- On cherche une dépendance non triviale  $X \rightarrow Y$  telle que  $R_i(X, Y, Z)$  et  $X$  n'est pas une clé de  $R_i$ .
- On ajoute à  $Y$  l'ensemble  $Z'$  des attributs de  $Z$  fonctionnellement déterminés par  $X$ , produisant la dépendance  $X \rightarrow Y \cup Z'$ .
- On remplace  $R_i$  dans  $S$  par les deux relations
  - $\underline{R_1(X, Y \cup Z')}$
  - $\underline{R_2(X, Z \setminus Z')}$

Propriété : cet algorithme est sans perte d'information mais pas toujours sans perte de dépendances

# Décomposition

Étant donnés un schéma de relation  $R$  et un ensemble  $F$  de DF, on va décomposer  $R$  en  $R_1, R_2, \dots, R_n$  tels que

- Cette décomposition soit **sans perte d'information** : on peut retrouver tout instance  $r$  de  $R$  en combinant *par jointure naturelle* les instances  $r_i$  sur les  $R_i$
- Cette décomposition soit **sans perte de dépendances** : après, jointure des  $r_i$ , on peut retrouver toutes les dépendances impliquées par  $F$  à partir de leurs projections  $F_i$  sur les  $R_i$
- $R_1, R_2, \dots, R_n$  soient dans une forme normale maximale

## Algorithme de décomposition et synthèse

- $R = \mathcal{U}$  la relation (universelle) à décomposer
- $F$  un *ensemble minimal* de dépendances sur  $R$

# Algorithme de décomposition

## Algorithme de décomposition en FNBC

Soit  $S = \{R\}$ . Tant qu'il existe dans  $R_i \in S$  qui n'est pas en FNBC :

- On cherche une dépendance non triviale  $X \rightarrow Y$  telle que  $R_i(X, Y, Z)$  et  $X$  n'est pas une clé de  $R_i$ .
- On ajoute à  $Y$  l'ensemble  $Z'$  des attributs de  $Z$  fonctionnellement déterminés par  $X$ , produisant la dépendance  $X \rightarrow Y \cup Z'$ .
- On remplace  $R_i$  dans  $S$  par les deux relations
  - $R_1(X, Y \cup Z')$
  - $R_2(X, Z \setminus Z')$

Propriété : cet algorithme est sans perte d'information mais pas toujours sans perte de dépendances

- Ces algorithmes d
- ces particulier : si on supprime  
mote on a une perte de points

# Algorithme de synthèse

## Algorithme de synthèse en 3FN/FNBC

- Générer une relation  $XY$  pour chaque DF  $X \rightarrow Y$ ;
- On supprime les schémas de relation qui ne sont pas maximaux par inclusion.
- S'il y a perte de jointure, alors on rajoute une relation composée d'une clé de  $F$ .

Propriété : l'algorithme est sans perte d'information et donne un schéma en 3FN ou en FNBC quand c'est possible sans perte de dépendance.

1 Problème et motivation : les anomalies

2 La théorie de la normalisation

3 La normalisation

- Les approches calculatoires
- Les diagramme Entités-Associations

4 Les contraintes d'intégrité en SQL

Autre démarche :

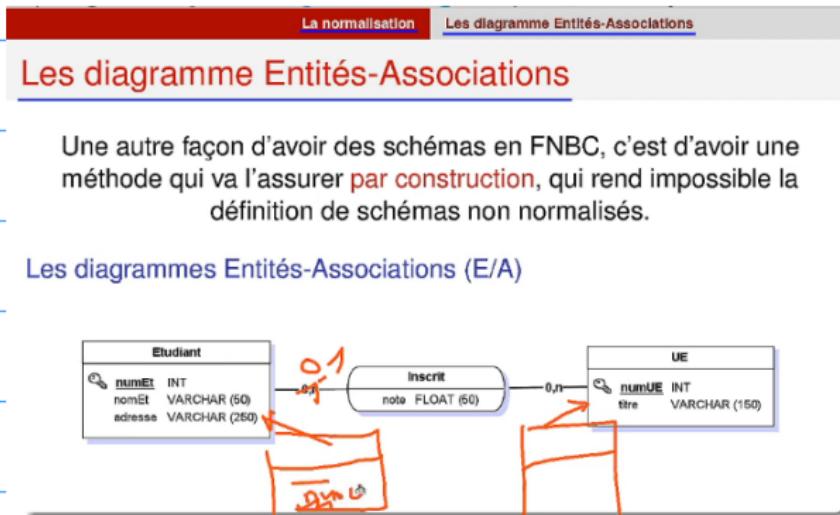
---

↳ de type conception

Journalisme en ligne / association  
plus restreint que le  
modèle relationnel

. Pour chaque entité on a nécessairement une clé constitutrice d'un

Passage du modèle entités-associations  
au modèle relationnel  
=> Laurent Audibert pages 75-82



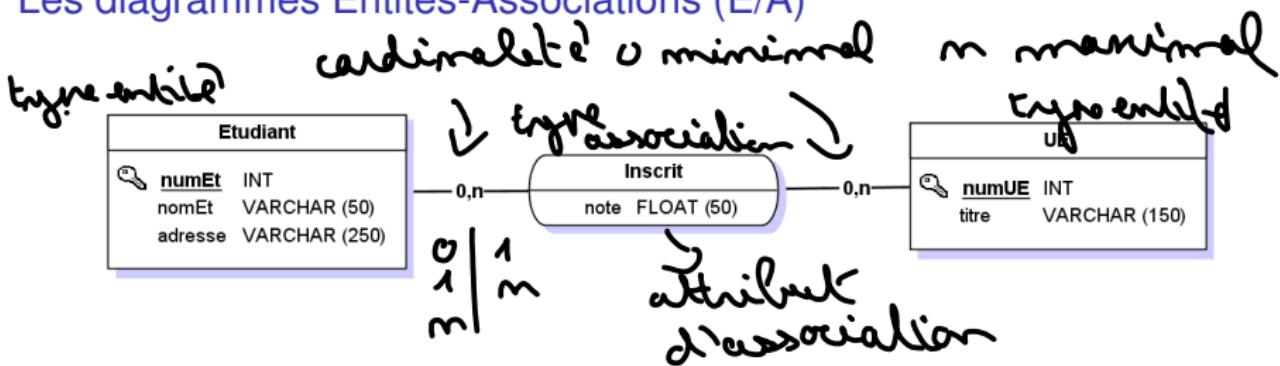
Voir par exemple

<https://perso.liris.cnrs.fr/fabien.duchateau/BDW1/>

# Les diagramme Entités-Associations

Une autre façon d'avoir des schémas en FNBC, c'est d'avoir une méthode qui va l'assurer **par construction**, qui rend impossible la définition de schémas non normalisés.

## Les diagrammes Entités-Associations (E/A)



Voir par exemple

<https://perso.liris.cnrs.fr/fabien.duchateau/BDW1/>

# Les diagramme Entités-Associations

## Méthode de conception avec E/A

- ① On définit d'abord un Modèle Conceptuel de Données (MCD) dans le formalisme E/A
  - On identifie les **entités** qui composent la base
  - On identifie les **associations** entre les entités, en précisant
    - les **cardinalités** de l'association
    - les attributs de l'association
- ② On génère le schéma logique SQL à partir du MCD
  - Pour chaque entité on crée une table
  - Pour chaque association, on étudie les cardinalités et selon
    - Cas *many-to-many* on génère une nouvelle table
    - Cas *many-to-one* on pousse l'association du côté du *one*

Voir par exemple JMerise <http://www.jfreesoft.com/JMerise/>

- 1 Problème et motivation : les anomalies
- 2 La théorie de la normalisation
- 3 La normalisation
- 4 Les contraintes d'intégrité en SQL

## Les contraintes d'intégrité en SQL

Le langage SQL permet de spécifier lors de la définition des schémas des contraintes sur les instances autorisées

### Principales contraintes d'intégrité

- NOT NULL : pas de valeur NULL pour l'attribut
  - Cette contrainte génère souvent automatiquement un index pour assurer efficacement la vérification de l'unicité
- UNIQUE : les valeurs (différentes de NULL) doivent être toutes différentes
- PRIMARY KEY : la même chose que UNIQUE et NOT NULL
- REFERENCES : clef étrangère, les valeurs prises doivent être présentes dans une colonne UNIQUE d'une autre table

Les contraintes **UNIQUE (NOT NULL)** permettent d'imposer les clefs dans les tables.

les clefs définissent les contraintes

# De la normalisation aux contraintes d'intégrités SQL

La normalisation permet d'obtenir des « bons schémas » (en FNBC) sans anomalies. Comment implanter cela en SQL ?

## Principe

- On fait l'hypothèse suivante **le concepteur de BD n'implémente que les clefs** (et les clés étrangères).
- Le contrôle automatique de ces contraintes par le SGBD est efficace et leur implémentation est toujours intégrée.
- Ainsi, toute mise-à-jour *respecte les clés*.

Par contre, on ne peut pas aussi facilement et aussi efficacement garantir les DFs qui ne sont pas des clefs !

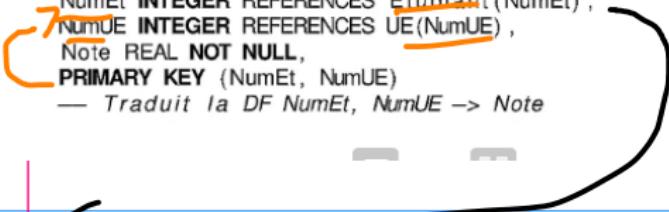
## Les contraintes d'intégrité en SQL

### La définition de la base d'exemple

```

CREATE TABLE Etudiant (
    NumEt INTEGER PRIMARY KEY,
    — Traduit la DF NumEt -> NomEt, Adresse
    | NomEt TEXT,
    | Adresse TEXT
);
CREATE TABLE UE (
    NumUE INTEGER PRIMARY KEY,
    — Traduit la DF NumUE -> Titre
    Titre TEXT
);
CREATE TABLE Inscrit (
    NumEt INTEGER REFERENCES Etudiant(NumEt),
    NumUE INTEGER REFERENCES UE(NumUE),
    Note REAL NOT NULL,
    PRIMARY KEY (NumEt, NumUE)
    — Traduit la DF NumEt, NumUE -> Note
);

```


 REFERENCES pour s'assurer que l'élément  
 Etudiant ou UE existe (et ce doit être un  
 attribut avec la contrainte Unique

pour des questions de performance  
(un attribut unique => indexé  
par la SGDB => performance  
pour retrouver les données)

# Les contraintes d'intégrité en SQL

Le langage SQL permet de spécifier lors de la définition des schémas des contraintes sur les instances autorisées

## Principales contraintes d'intégrité

- NOT NULL : pas de valeur NULL pour l'attribut
- UNIQUE : les valeurs (différentes de NULL) doivent être toutes différentes
  - Cette contrainte génère automatiquement un index pour assurer efficacement la vérification de l'unicité
- PRIMARY KEY : la même chose que UNIQUE et NOT NULL
- REFERENCES : clef étrangère, les valeurs prises doivent être présentes dans une colonne UNIQUE d'une autre table

Les contraintes UNIQUE (NOT NULL) permettent d'imposer les clefs dans les tables.

# Les contraintes d'intégrité en SQL

## La définition de la base d'exemple

```
CREATE TABLE Etudiant (
    NumEt INTEGER PRIMARY KEY,
    — Traduit la DF NumEt → NomEt, Adresse
    NomEt TEXT,
    Adresse TEXT
);
CREATE TABLE UE (
    NumUE INTEGER PRIMARY KEY,
    — Traduit la DF NumUE → Titre
    Titre TEXT
);
CREATE TABLE Inscrit (
    NumEt INTEGER REFERENCES Etudiant(NumEt) ,
    NumUE INTEGER REFERENCES UE(NumUE) ,
    Note REAL NOT NULL,
    PRIMARY KEY (NumEt, NumUE)
    — Traduit la DF NumEt, NumUE → Note
);
```