

# Chapitre 3

---

## *Modèle relationnel*

# Modèle relationnel

---

Les points abordés dans ce chapitre sont les suivants :

- *Définition du modèle relationnel*
- *Transformation des diagrammes E-R en tableaux relationnels*
- *Relations statiques et dynamiques*
- *Dépendance fonctionnelle*
- *Normalisation et dénormalisation*
- *Exemple de conception d'un schéma relationnel*

# Définition du modèle relationnel

---

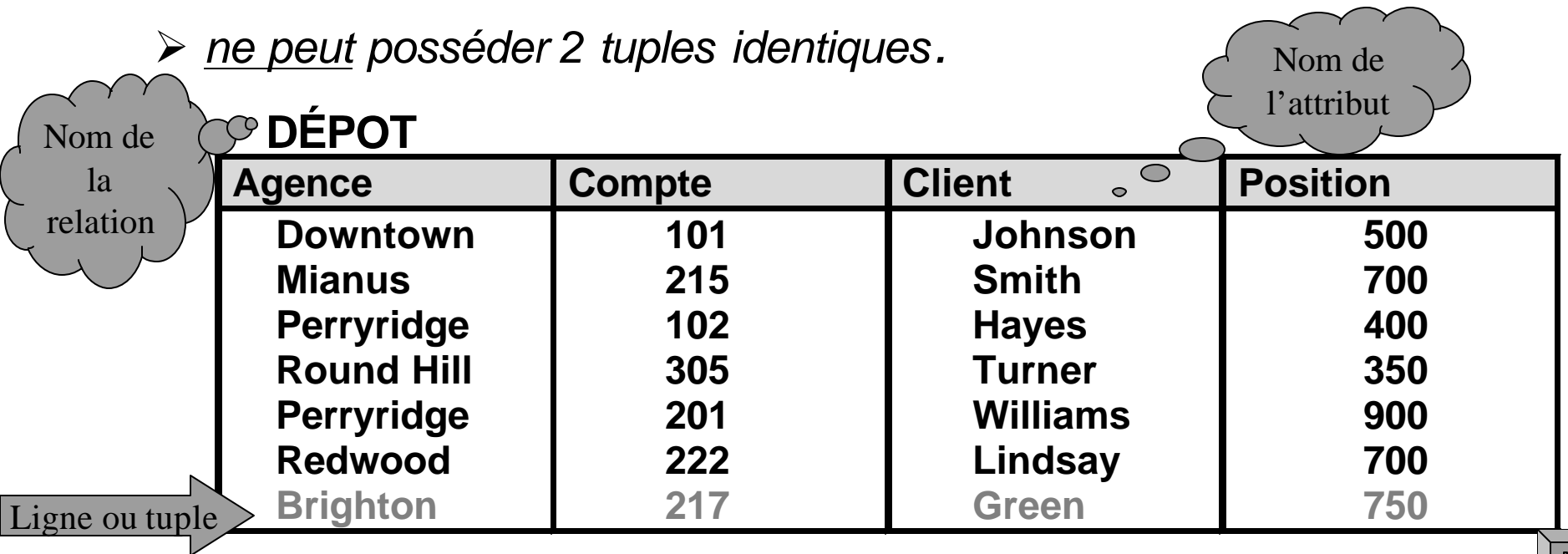
## Caractéristiques:

- *Développé par Codd en 1970: théorie mathématique des relations.*
- *Modèle logique orienté enregistrement.*
- *Constitué de tableaux appelés relations.*
- *Simple: la relation est la seule structure du modèle.*
- *La définition de relation est plus générale qu'au modèle E-R.*

# Définition du modèle relationnel

## Relation

- *représentée par un tableau à 2 dimensions*
- *composée d'un nombre fini de colonnes (attributs)*
- *chaque attribut possède un nom unique à l'intérieur d'une relation.*
- *ne peut posséder 2 tuples identiques.*



The diagram shows a table representing a relation. Annotations include:

- A cloud bubble pointing to the table title: "Nom de la relation"
- A cloud bubble pointing to a column header: "Nom de l'attribut"
- An arrow pointing to a row: "Ligne ou tuple"

Agence	Compte	Client	Position
Downtown	101	Johnson	500
Mianus	215	Smith	700
Perryridge	102	Hayes	400
Round Hill	305	Turner	350
Perryridge	201	Williams	900
Redwood	222	Lindsay	700
Brighton	217	Green	750

# Définition du modèle relationnel

---

Chaque attribut possède un domaine de valeurs  $(D_1, D_2, \dots, D_n)$ .

Exemples:

NAS : ensemble des nombre de 9 digits de sécurité sociale, valides.

Nom : ensemble des noms de personnes.

Age : âge possible : 16-70

# de téléphone : (ddd)ddd-dddd

Un tuple est composé de  $n$  éléments :

$(v_1, v_2, \dots, v_n)$  où  $v_1 \in D_1, v_2 \in D_2, \dots, v_n \in D_n$

Un tableau est un sous-ensemble du produit cartésien d'un ensemble de domaines.

$$R = \prod_{i=1}^n D_i$$

# Définition du modèle relationnel

---

## Base de données relationnelle (BDR)

- *regroupement d'un ensemble de relations.*
- *chaque relation est nommée de façon unique.*
- représentée par le ***schéma relationnel*** ou le ***diagramme du schéma relationnel***.

# Définition du modèle relationnel

---

## Schéma relationnel

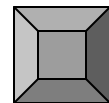
➤ *Composé du nom et de la liste des attributs d'une relation.*

***R (A1, A2, ..., An) où***

- R est le nom de la relation.
- Ai : Attribut (rôle joué par un certain domaine dans le schéma de relation R).

*Exemple:*

*Dépôt(agence, compte, client, position)*



# Définition du modèle relationnel

---

## Diagramme du schéma relationnel

- *Composé du nom et de la liste des attributs d'une relation présenté de façon graphique.*

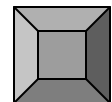
*Nom de la relation*

<b>A1</b>	<b>A2</b>	<b>...</b>	<b>An</b>
-----------	-----------	------------	-----------

*Exemple:*

*Dépôt*

<b>Agence</b>	<b><u>Compte</u></b>	<b>Client</b>	<b>Position</b>
---------------	----------------------	---------------	-----------------



# Définition du modèle relationnel

---

## *Autre exemple*

Client	Rue	Localité
Jones	Main	Harrison
Smith	North	Rye
Hayes	Main	Harrison
Curry	North	Rye
Lindsay	Park	Pittsfield
Turner	Putnam	Stamford
Williams	Nassau	Princeton
Adams	Spring	Pittsfield
Johnson	Alma	Palo Alto
Glenn	Sand Hill	Woodside
Brooks	Senator	Brookside
Green	Walnut	Stamford

## La relation Clientèle

Schéma relationnel : Clientèle( client, rue, localité )

Diagramme du schéma relationnel:

Clientèle

<u>Client</u>	Rue	Localité
---------------	-----	----------

# Définition du modèle relationnel

---

## Degré de la relation

➤ *nombre d'attributs (n) dans son schéma relationnel*

**Ex:**

Cette relation est de degré 7 :

*ÉTUDIANT( Nom, NAS, Téléphone, Adresse,  
TéléphoneBureau, Âge, Département)*

Donc:

La relation Dépôt est de degré 4.

La relation Clientèle est de degré 3.

# Définition du modèle relationnel

---

## Contraintes relationnelles

### ➤ *4 types de contraintes*

- de domaines
- de clés
- d'intégrité
- d'intégrité référentiel

### ➤ *doivent être vérifiées par chaque instance du schéma*

## Contraintes de domaines

### ➤ La valeur de chaque attribut dans un tuple est atomique. (non divisible)



*Attributs composites ou multivalués*

### ➤ La valeur doit respecter le format des données du domaine de l'attribut (entier, réel, date, caractère)

# Définition du modèle relationnel

---

## Contraintes de clés

- *Chaque tuple dans une relation doit être unique.*
- *Toute relation doit posséder une clé qui identifie un tuple de façon unique.*
- *Une relation peut posséder plusieurs clés candidates.*
  - À la limite, l'ensemble de tous les attributs constitue une clé.
- *La clé choisie est appelé **clé primaire**.*
  - Elle est soulignée dans la relation.

*Exemple:*

*ÉTUDIANT(Nom,NAS,CodePermanent,Adresse,Téléphone,DateNaissance)*

# Définition du modèle relationnel

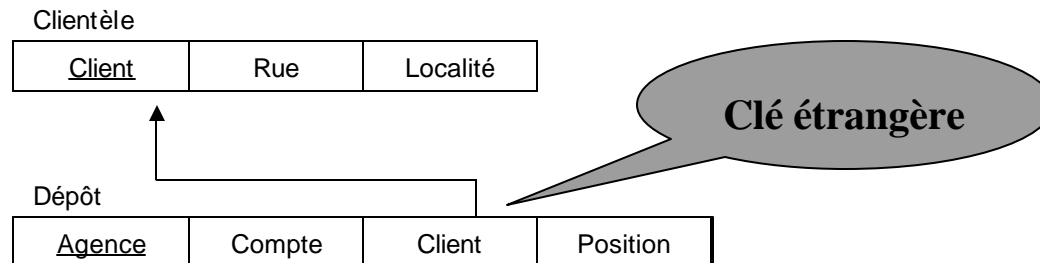
---

## Contraintes d'intégrité-entité

- stipulent qu'**aucune** clé primaire ne doit être nulle.

## Contraintes d'intégrité de référence

- contraintes spécifiées entre deux relations et utilisées pour maintenir la consistance entre les tuples de deux relations.
- concept de clé étrangère ( Foreign Key ).



# Avantage du modèle relationnel

---

Exemple:

Question : Pourquoi ne pas créer un seul schéma relationnel (Compte) qui englobe tous les attributs?

*Compte (agence, compte, client, position, rue, ville )*

Réponse : Ceci occasionne les problèmes suivants :

- ***une perte d'espace mémoire***
  - réplication de l'information rue et ville pour tous les comptes d'un client
- ***une difficulté de mise à jour:***
  - Pour un changement d'adresse, il faut changer l'information pour chaque compte au lieu d'un seul tuple dans la relation Clientèle.
- ***si une information est manquante, il est impossible de créer **Compte** (à moins d'ajouter des valeurs nulles pour rue et ville).***

# Avantage du modèle relationnel

---

Voilà pourquoi il est plus avantageux d'utiliser deux relations :

Dépôt (agence, compte, client, position)

Clientèle (client, rue, ville )

- *Économie d'espace*
- *Cohérence des données*

Si le client désire ouvrir un compte mais n'a pas d'adresse fixe pour le moment, le modèle relationnel permet :

- *de créer d'un tuple dans **Dépôt***
- *sans créer de tuple dans **Clientèle**.*

# Transformation des diagrammes E-R en tableaux relationnels

---

Puisque le modèle E-R est un modèle objet, il faut retrouver le triplet Objets-Attributs-Valeurs dans les tableaux relationnels.

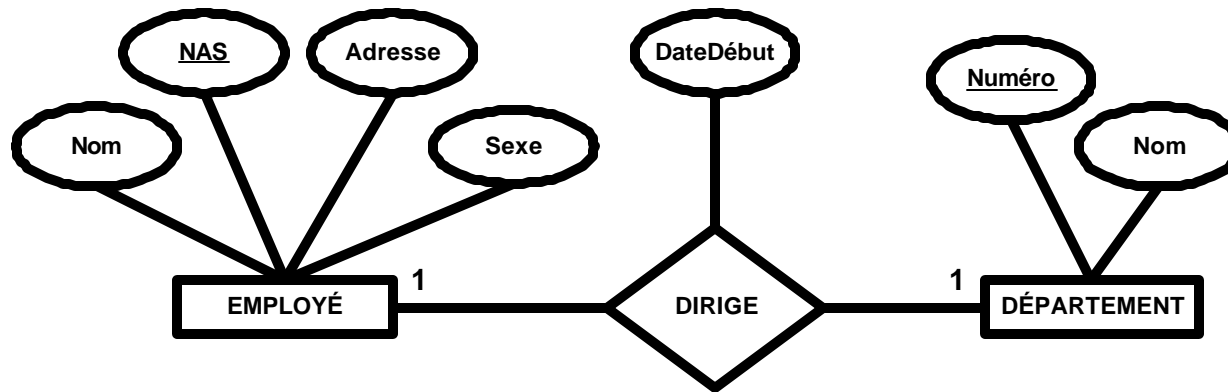
- **Objets** ® Entités ® Noms des tableaux (fichiers, DDL)
- **Attributs** ® Attributs ® Schémas des tableaux (les variables, DDL)
- **Valeurs** ® Valeurs ® Instanciations des tableaux. (données, DML)

Règles de base:

- Chaque type d'entités devient une relation avec les mêmes attributs.
- On inclut seulement les attributs simples des attributs composites.
- Clé primaire d'une entité = la clé primaire de la relation.
- Les types d'entités faibles subissent la même transformation, mais il faut ajouter une clé étrangère qui correspond à la clé primaire de l'entité forte.
- Chaque type de relations subit la transformation relié à sa cardinalité.

# Transformation des diagrammes E-R en tableaux relationnels

Pour les relations 1 vers 1 :



➤ inclure la clé primaire d'une entité dans l'autre entité **ou** combiner les deux entités,

Employé

Nom	<u>NAS</u>	Adresse	Sexe	NuméroD	NomD	DateDébut
-----	------------	---------	------	---------	------	-----------

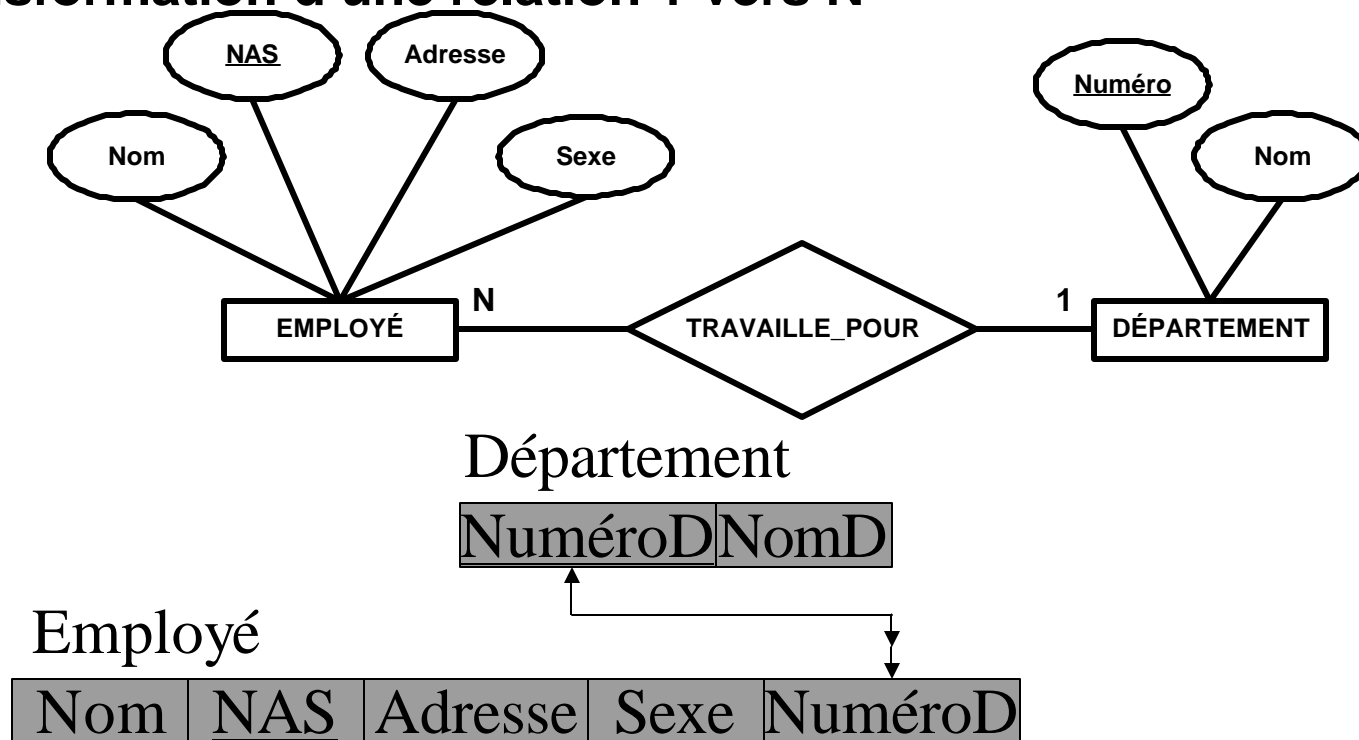
*Tableau (autre exemple)*

# Transformation des diagrammes E-R en tableaux relationnels

## Pour les relations 1 vers n et n vers 1

➤ Inclure la clé primaire de l'entité de cardinalité 1 dans l'autre entité de cardinalité n.

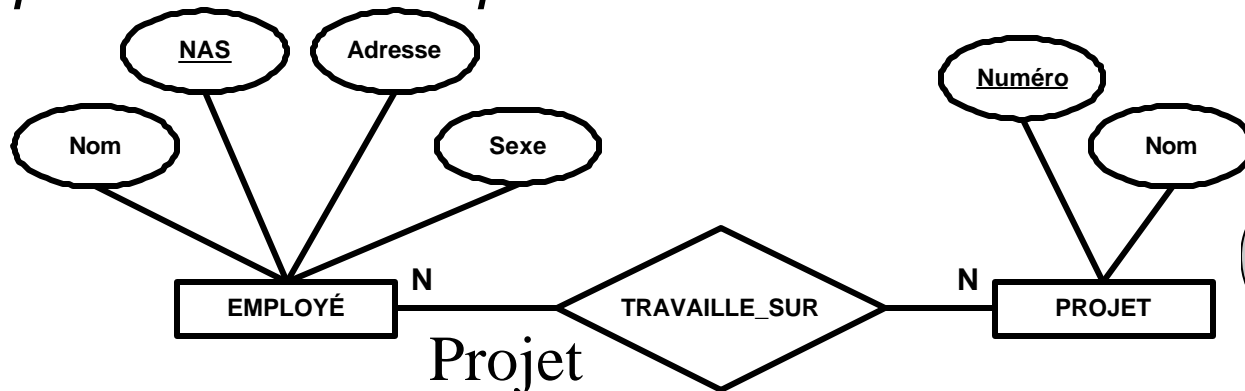
### Transformation d'une relation 1 vers N



# Transformation des diagrammes E-R en tableaux relationnels

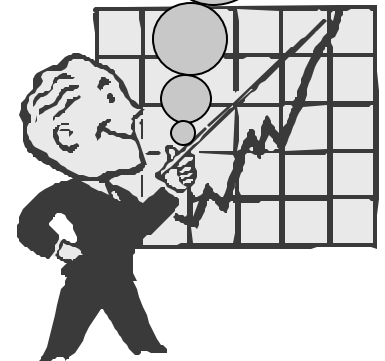
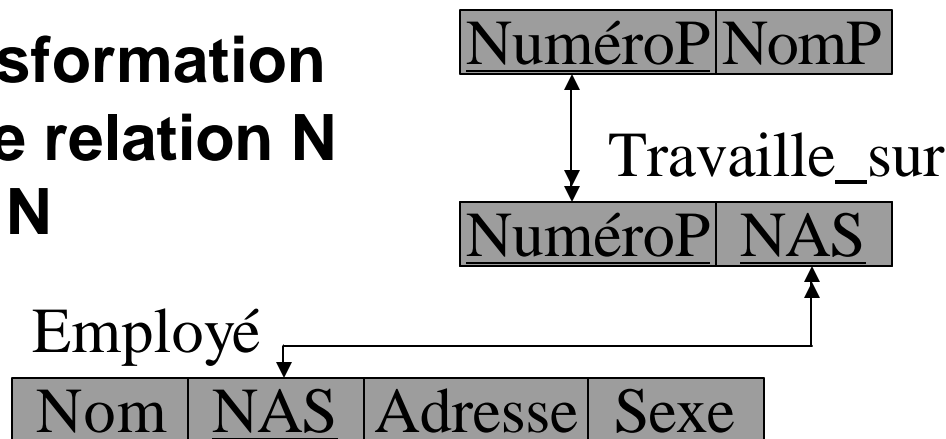
## Pour les relations n vers n

- *Il faut créer le tableau de la relation et inclure la clé primaire de chaque entité dans ce tableau.*



Exemple de transformation E-R en schéma relationnel. (Service de Livraison)

## Transformation d'une relation N vers N



# Relations statiques et dynamiques

---

Deux types de relations :

- *Les relations statiques*
- *les relations dynamiques*

Relations statiques

- *Leur **dimension** et leur **contenu** ont tendances à se stabiliser avec le temps.*
- *Les données varient à l'occasion, mais de façon générale reste stable.*

Relations dynamiques

- *Elle font vivre la base de données et représentent la plupart du temps les interactions entre les relations statiques.*

# Relations statiques et dynamiques

## Exemple:

La relation entre S et P est de cardinalité **n vers n** (un constructeur fabrique plusieurs pièces et une pièce est fabriquée par différents constructeurs).

Constructeur

S

SNO	SNAME	STATUS	CITY
S1	Smith	20	London
S2	Jones	10	Paris
S3	Blake	30	Paris
S4	Clark	20	London
S5	Adams	30	Athens

Relations  
Statiques

P

PNO	PNAME	COLOR	WEIGHT	CITY
P1	Nut	Red	12	London
P2	Bolt	Green	17	Paris
P3	Screw	Blue	17	Rome
P4	Screw	Red	14	London
P5	Cam	Blue	12	Paris
P6	Cog	Red	19	London

Pièce

SP

SNO	PNO	QTY
S1	P1	300
S1	P2	200
S1	P3	400
S1	P4	200
S1	P5	100
S1	P6	100
S2	P1	300
S2	P2	400
S3	P2	200
S4	P2	200
S4	P4	300
S4	P5	400

Commande

Relation  
Dynamique

# Dépendance fonctionnelle

---

Le modèle relationnel permet de structurer les données à stocker dans la BD.

ex: même situation  $\nabla$  différents schémas relationnels

Les SGBD-R sont plus efficaces si:

- *Les requêtes sont faciles à écrire.*
- *Les données sont faciles à accéder.*
- *L'intégrité des données est conservée (mises à jour).*

Leur objectif est d'éviter tout problème d'ajout, de suppression ou de mise à jour de données. Il faut normaliser les données.

Théorie de la normalisation

- *basée sur la notion de dépendance fonctionnelle*

# Dépendance fonctionnelle

## Notion de dépendance entre attributs

- *La valeur de la clé pour tout tuple donné doit déterminer les valeurs pour tous les autres attributs du tuple.*

**Ex :**      *Information sur les réclamations d'une compagnie d'assurances*

*Réclamation*

<u>Numéro</u>	Client	Adresse
101	Marie	Montréal
102	Steve	Toronto
103	Marie	Montréal

Répétition de  
l'adresse



# Dépendance fonctionnelle

---

## Exemple (suite):

Décomposition en deux relations à l'aide de la clé primaire client.

<u>Numéro</u>	Client
101	Marie
102	Steve
103	Marie

<u>Client</u>	Adresse
Marie	Montréal
Steve	Toronto

**Plus de  
répétition**



À partir du nom du client, on obtient:

- *son adresse*
- *le numéro de ses réclamations*

On dit qu'il y a dépendance entre l'attribut client et les autres attributs.

# Dépendance fonctionnelle

---

## Contrainte de dépendance fonctionnelle

- *permet de normaliser une BD;*
- *représente la généralisation du concept de clé.*

Une dépendance fonctionnelle est notée :

$X \twoheadrightarrow Y$ , où  $X$  et  $Y$  sont des attributs de  $R$

$X \twoheadrightarrow Y$  signifie que si on connaît la valeur de  $X$ , on peut connaître celle de  $Y$  ( $X$  détermine  $Y$  ou  $Y$  a une dépendance fonctionnelle de  $X$ )

$K$  est une clé candidate de  $R$  si  $K \rightarrow R$  (tous les attributs de la relation)

i.e. il n'existe aucun couple de tuples présentant les mêmes valeurs sur l'ensemble d'attributs  $K$ .

# Dépendance fonctionnelle

Exemple 1 :

En examinant l'instanciation de la relation R, on peut y découvrir quelques dépendances fonctionnelles.

**Relation R :**

A	B	C	D
a1	b1	c1	d1
a1	b2	c1	d2
a2	b2	c2	d2
a2	b3	c2	d3
a3	b3	c2	d4

Pour la même clé, nous aurons la même valeur associée.

**$A \twoheadrightarrow A$  : triviale,**

**$A \twoheadrightarrow C$  : est vérifiée,**

**$AB \twoheadrightarrow D$  : pas certain (pas 2 clés pareils),**

**$AC \twoheadrightarrow D$  : n'est pas vérifiée,**

**et**

**$C \twoheadrightarrow A$  : n'est pas vérifiée !**

# Dépendance fonctionnelle

Exemple 2 :

Pour une agence donnée, elle sera située dans une ville unique et aura un avoir unique.

Agence	Avoirs	Ville	Prêt	Client	Montant
Downtown	9000000	Brooklyn	17	John	1000
Redwood	2100000	Palo Alto	23	Smith	2000
Perryridge	1700000	Horseneck	15	Hayes	1500
Downtown	9000000	Brooklyn	14	Jackson	1500
Mianus	400000	Horseneck	93	Curry	500
Round Hill	8000000	Horseneck	11	Turner	900

Contraintes imposées par le concepteur qui force l'attribut Agence comme clé primaire

**Agence** ® **Ville**

**Agence** ® **Avoirs**

# Dépendance fonctionnelle

*Exemple 3 :*

- *dépendance  $\text{rue} \rightarrow \text{localité}$  est vérifiée,*
- *mais invalide, on peut supposer que plusieurs localités peuvent avoir une rue Main.*

Client	Rue	Localité
Jones	Main	Harrison
Smith	North	Rye
Hayes	Main	Harrison
Curry	North	Rye
Lindsay	Park	Pittsfield
Turner	Putnam	Stamford
Williams	Nassau	Princeton
Adams	Spring	Pittsfiel
Johnson	Alma	Palo Alto
Glenn	Sand Hill	Woodside
Brooks	Senator	Brooklyn
Green	Walnut	Stamford

# Dépendance fonctionnelle

---

*Exemple 4 :*

Soit la relation

Étudiant(CodePermanent, NAS, Nom)

Il existe

CodePermanent  $\rightarrow$  NAS, Nom

Mais aussi

NAS  $\rightarrow$  CodePermanent, Nom

Exercice chapitre 3 #3 et autre exemples p.66, p.84

# Dépendance fonctionnelle

---

## Propriétés des dépendances fonctionnelles (DF)

- *permettent de définir la fermeture  $F^+$  d'un ensemble initial de dépendances fonctionnelles  $F$ .*

### Propriétés de base

#### Réflexivité

- *Si  $Y$  est inclus dans  $X$  ( $Y \dot{\subseteq} X$ ), alors  $X \twoheadrightarrow Y$*

#### Augmentation

- *Si  $X \twoheadrightarrow Y$  et  $W$  est un ensemble d'attributs, alors nous pouvons dire  $WX \twoheadrightarrow WY$*

#### Transitivité

- *Si  $X \twoheadrightarrow Y$  et  $Y \twoheadrightarrow Z$ , alors  $X \twoheadrightarrow Z$*

# Dépendance fonctionnelle

---

*Les trois prochaines propriétés découlent des trois premières*

## Pseudo transitivité

➤ *Si  $X \textcircled{R} Y$  et  $YW \textcircled{R} Z$ , alors  $XW \textcircled{R} Z$*

*Démonstration: 1.  $X \textcircled{R} Y$  par augmentation  $XW \textcircled{R} YW$*

*2.  $XW \textcircled{R} YW$ ,  $YW \textcircled{R} Z$  par transitivité  $XW \textcircled{R} Z$*

## Union

➤ *Si  $X \textcircled{R} Y$  et  $X \textcircled{R} Z$ , alors  $X \textcircled{R} YZ$*

*Démonstration: 1.  $X \textcircled{R} Y$  par augmentation  $XZ \textcircled{R} YZ$*

*2.  $XZ \textcircled{R} YZ$ ,  $X \textcircled{R} Z$  par pseudo-transitivité  $X \textcircled{R} YZ$*

# Dépendance fonctionnelle

---

## Décomposition

➤ Si  $X \twoheadrightarrow YZ$ , alors  $X \twoheadrightarrow Y$  et  $X \twoheadrightarrow Z$

*Démonstration:*

1. D'après la réflexivité  $YZ \twoheadrightarrow Y$  et  $YZ \twoheadrightarrow Z$

2.  $X \twoheadrightarrow YZ$  et  $YZ \twoheadrightarrow Y$  par transitivité  $X \twoheadrightarrow Y$

3.  $X \twoheadrightarrow YZ$  et  $YZ \twoheadrightarrow Z$  par transitivité  $X \twoheadrightarrow Z$

# Fermeture d'un ensemble de dépendances fonctionnelles

R (A, B, C, G, H, I)

F :  $A \rightarrow B$

$A \rightarrow C$

$B \rightarrow H$

$CG \rightarrow H$

$CG \rightarrow I$

Schéma  
relationnel

Dépendances  
fonctionnelles

$F^+$  : fermeture de F

- l'ensemble de toutes les dépendances fonctionnelles logiquement impliquées par F.

$F^+$ :

$A \rightarrow H$  : par transitivité

$A \circledast B$  et  $B \circledast H = A \circledast H$

$CG \rightarrow HI$  : par union

$CG \circledast H$  et  $CG \circledast I$

$AG \rightarrow I$  : par deux façons

1.  $A \rightarrow C$  par augmentation devient  $AG \rightarrow CG$ .  $AG \rightarrow CG$  et  $CG \rightarrow I$  par transitivité devient  $AG \rightarrow I$
2.  $A \rightarrow C$  et  $CG \rightarrow I$  par pseudo-transitivité devient  $AG \rightarrow I$

Quelques  
éléments de  
fermeture

# Fermeture d'un ensemble de dépendances fonctionnelles

---

**Exemple :** Calculer des éléments de la fermeture  $F^+$  du schéma relationnel suivant:

$R = (A, B, C, D, E)$

où  $F$  :

- $A \rightarrow BC$

- $CD \rightarrow E$

- $B \rightarrow D$

- $E \rightarrow A$

Trouver quelles sont les clés primaires?

1. De  $A \twoheadrightarrow BC$  on obtient par décomposition :  $A \twoheadrightarrow B$ ,  $A \twoheadrightarrow C$
2. Puisque  $A \twoheadrightarrow B$  et  $B \twoheadrightarrow D$  on obtient par transitivité :  $A \twoheadrightarrow D$
3. Puisque  $A \twoheadrightarrow C$  et  $A \twoheadrightarrow D$  on obtient par union :  $A \twoheadrightarrow CD$

# Fermeture d'un ensemble de dépendances fonctionnelles

---

4. Puisque  $A \rightarrow CD$  et  $CD \rightarrow E$   
on obtient par transitivité :  $A \rightarrow E$
  5. Puisque  $A \rightarrow A$  (triviale) et  $A \rightarrow B$ ,  $A \rightarrow C$ ,  $A \rightarrow D$  et  $A \rightarrow E$ ,  
on obtient par union :  $A \rightarrow ABCDE$
  6. Puisque  $E \rightarrow A$  et  $A \rightarrow ABCDE$   
on obtient par transitivité :  $E \rightarrow ABCDE$
  7. Puisque  $CD \rightarrow E$  et  $E \rightarrow ABCDE$   
on obtient par transitivité :  $CD \rightarrow ABCDE$
  8. Puisque  $B \rightarrow D$   
on obtient par augmentation :  $BC \rightarrow CD$
  9. Puisque  $BC \rightarrow CD$  et  $CD \rightarrow ABCDE$   
on obtient par transitivité :  $BC \rightarrow ABCDE$
- *Clés privilégiées ou candidates : A, E, CD, BC*

# Décomposition d'une base de données sans pertes

---

Le concepteur impose les dépendances fonctionnelles lors de l'élaboration d'une base de données relationnelles.

La théorie de la **normalisation** consiste à repérer ces dépendances et à séparer les relations (si nécessaire) en sous-relations où les anomalies seront absentes.

Pour ce faire, il faut s'assurer que la décomposition est **sans perte (jonction conservatrice)** en respectant ce critère:

- *Soit  $R$ , un schéma relationnel,  $F$  un ensemble de dépendances fonctionnelles, et  $R1$  et  $R2$  une décomposition  $R$ ,*
  - Cette décomposition est sans perte si au moins l'une des dépendances fonctionnelles suivantes est dans  $F^+$  :
    - ♦  $R1 \Join R2 \stackrel{R}{\neq} R1$
    - ♦  $R1 \Join R2 \stackrel{R}{\neq} R2$

# Décomposition d'une base de données sans pertes

---

## Exemple :

Voici les schémas et leurs dépendances fonctionnelles imposées par le concepteur pour une BD d'une banque.

### Schémas

Banque (agence, ville, avoirs)

Clientèle (client, rue, localité)

Crédit (agence, prêt, client, montant)

Dépôt (agence, compte client, position)

### Dépendances fonctionnelles

Agence ® ville

Agence ® avoirs

Client ® rue

Client ® localité

Prêt ® montant

Prêt ® agence

Compte ® position

Compte ® agence

# Décomposition d'une base de données sans pertes

---

Le schéma suivant comporte des problèmes importants:

*Emprunt (agence, avoirs, ville, prêt, client, montant)*

L'ensemble F des dépendances fonctionnelles imposées par le concepteur sont les suivantes :

- *agence ® avoirs ( une agence a un seul avoir)*
- *agence ® ville ( une agence se situe dans une seule ville mais une ville peut contenir plusieurs agences )*
- *prêt ® montant (un seul numéro de prêt par montant prêté)*
- *prêt ® agence (un numéro de prêt ne se retrouve que dans une agence)*

On veut décomposer la relation Emprunt de sorte qu'elle soit à jonction conservatrice.

# Décomposition d'une base de données sans pertes

---

1. À l'étape initiale, on décompose Emprunt en 2 sous-schémas :

$R1 = (agence, avoirs)$  et  $R1' = (agence, ville, prêt, client, montant)$

Comme  $R1 \subset R1' = \{agence\}$ , est-ce que  $R1 \subset R1' \text{ @ } R1$  ?

➤  $Agence \text{ @ } \{agence, avoirs\}$

Preuve :

on a  $agence \rightarrow avoirs$  dans  $F$ , et par augmentation on obtient:

$agence, agence \text{ @ } agence, avoirs$

$agence \text{ @ } \{agence, avoirs\}$

➤  $R1$  est à jonction conservatrice !

# Décomposition d'une base de données sans pertes

---

2. on décompose l'ensemble restant (ici  $R1'$ ) en 2 sous-schémas:

$R2 = (\text{agence}, \text{ville})$  et  $R2' = (\text{agence}, \text{prêt}, \text{client}, \text{montant})$

Comme  $R2 \cap R2' = \{\text{agence}\}$ , est-ce que  $R2 \cap R2' \rightarrow R2$  ?

➤ *Agence  $\textcircled{R}$  {agence, ville}*

Preuve :

on a  $\text{agence} \rightarrow \text{ville}$  dans  $F$ , et par augmentation on obtient:

*agence, agence  $\textcircled{R}$  agence, ville*

*agence  $\textcircled{R}$  {agence, ville}*

➤  *$R2$  est à jonction conservatrice !*

# Décomposition d'une base de données sans pertes

---

3. Ensuite, on décompose l'ensemble restant (ici  $R_2'$ ) en 2 sous-schémas:

$R_3 = (\text{prêt}, \text{montant})$  et  $R_3' = (\text{prêt}, \text{agence}, \text{client})$

Comme  $R_3 \subset R_3' = \{\text{prêt}\}$ , est-ce que  $R_3 \subset R_3' \text{ @ } R_3$  ?

➤  $\text{prêt @ } \{\text{prêt}, \text{montant}\}$

Preuve :

on a  $\text{prêt} \rightarrow \text{montant}$  dans  $F$ , et par augmentation on obtient:

$\text{prêt}, \text{prêt} \text{ @ } \text{prêt}, \text{montant}$

$\text{prêt @ } \{\text{prêt}, \text{montant}\},$

➤  $R_3$  est à jonction conservatrice !

# Décomposition d'une base de données sans pertes

---

4. Ensuite, on décompose l'ensemble restant (ici  $R3'$ ) en 2 sous-schémas:

$R4 = (\text{prêt}, \text{agence})$  et  $R4' = (\text{prêt}, \text{client})$

Comme  $R4 \subset R4' = \{\text{prêt}\}$ , est-ce que  $R4 \subset R4' \text{ @ } R4$  ?

➤  $\text{prêt @ } \{\text{prêt}, \text{agence}\}$

Preuve :

on a  $\text{prêt} \rightarrow \text{agence}$  dans  $F$ , et par augmentation on obtient:

$\text{prêt}, \text{prêt @ prêt}, \text{agence}$

$\text{prêt @ } \{\text{prêt}, \text{agence}\},$

➤  $R4$  est à jonction conservatrice !

# Décomposition d'une base de données sans pertes

---

Finalement on obtient le schéma Emprunt décomposé en 5 relations sans pertes où  $R5 = R4'$ :

- $R1$  (*agence, avoirs*)
- $R2$  (*agence, ville*)
- $R3$  (*prêt, montant*)
- $R4$  (*prêt, agence*)
- $R5$  (*prêt, client*)

# Normalisation

---

La méthode consiste à créer des schémas relationnels répondant à un certain standard appelé **forme normale** qui devront respecter les **contraintes de dépendances**.

## La normalisation permet...

- *de guider la structuration des schémas relationnels,*
- *de vérifier qu'une décomposition est à jonction conservatrice,*
- *d'imposer des contraintes aux relations d'une BD.*

## Afin ...

- *de rendre une BD la plus efficace possible et sans redondances inutiles.*

# Normalisation

---

Cette procédure fait subir à une relation une série de tests pour certifier qu'elle appartient à une certaine forme normale.

Codd (mathématicien, chercheur de IBM, 1972) a proposé trois formes normales:

- ◆ *1NF : première forme normale,*
- ◆ *2NF : deuxième forme normale,*
- ◆ *3NF : troisième forme normale.*

Ces formes normales sont basées sur les dépendances fonctionnelles entre les attributs d'une relation.

# Normalisation

---

Avec le temps, d'autres formes normales se sont développées:

- ◆ *BCNF* : forme normale de Boyce-Codd
- ◆ *4NF* : quatrième forme normale
- ◆ *5NF* : cinquième forme normale
- ◆ *PJNF* : forme normale à projections jointives
- ◆ *DKN* : forme normale à domaines-clés

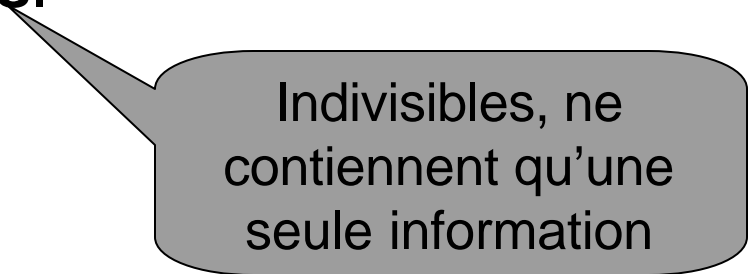
Elles traitent des cas particuliers non résolus par l'application successive des trois premières formes normales.

# 1NF : Première forme normale

---

La première forme normale contient 2 règles:

**1. R** est une relation **1NF** ssi (si et seulement si ) les domaines de la relation contiennent uniquement des **valeurs atomiques**.



Indivisibles, ne  
contiennent qu'une  
seule information



*Les attributs à valeurs multiples (multivalués) ne sont pas permis. !*

# 1NF : Première forme normale

*Exemple de valeur non atomique :*

L'attribut **DLOCATIONS** de l'exemple suivant est un exemple de valeur non-atomique.

a) **DEPARTEMENT**

DNAME	<u>DNUMBER</u>	DMGRSSN	DLOCATIONS
			...

Valeurs non atomiques

**a) Relation qui n'est pas 1NF.**

b) **DEPARTEMENT**

DNAME	<u>DNUMBER</u>	DMGRSSN	DLOCATIONS
Research	5	333445555	(Bellaire,Sugarland,Houston)
Administration	4	987654321	(Stafford)
Headquarters	1	888665555	(Houston)

**b) Instanciation de la relation.**

c) **DEPARTEMENT**

DNAME	<u>DNUMBER</u>	DMGRSSN	<u>DLOCATION</u>
Research	5	333445555	Bellaire
Research	5	333445555	Sugarland
Research	5	333445555	Houston
Administration	4	987654321	Stafford
Headquarters	1	888665555	Houston

**c) Relation 1NF avec redondance**

# 1NF : Première forme normale

2. R est une relation **1NF** ssi les domaines de la relation ne contiennent pas de relation.

♦ *pour chacune des positions colonne d'une relation, il n'y a pas de relation imbriquée.*

 Les attributs composites ne sont pas permis !

Exemple:

a) PROJS est inclus dans EMP\_PROJ

b) Instanciation

c) EMP\_PROJ décomposé

a) EMP\_PROJ

SSN	ENAME	PROJS	
		PNUMBER	HOURS

b) EMP\_PROJ

SSN	ENAME	PNUMBER	HOURS
12345678	Smith, John B.	1	32.5
		2	7.5
666884444	Narayan, Ramesh K.	3	40.0
453453453	English, Joyce A.	1	20.0
		2	20.0
333445555	Wong, Franklin T.	2	10.0
		3	10.0
		10	10.0
		20	10.0
999887777	Zelaya, Alicia J.	30	30.0
987987987	Jabbar, Ahmad V.	10	10.0
		30	5.0
987654321	Wallace, Jennifer S.	30	20.0
		20	15.0
888665555	Borg, James E.	20	null

c) EMP\_PROJ1

SSN	ENAME
-----	-------

EMP\_PROJ2

SSN	PNUMBER	HOURS
-----	---------	-------

# 2NF: Deuxième forme normale

---

R est une relation **2NF** ssi :

1. *elle est 1NF,*
2. *tous les attributs “non clés” sont complètement dépendants de la clé primaire (mais peuvent l’être transitivement).*

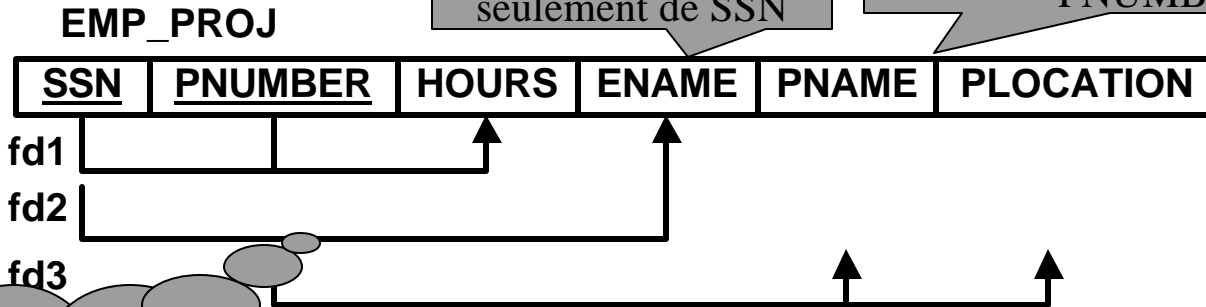
*i.e. il ne peuvent pas être dépendant de seulement une partie de la clé. Toute la clé est nécessaire pour retrouver l’attribut “non clé”.*

La **2NF** est basée sur le concept de dépendance fonctionnelle *complète*.

Une dépendance fonctionnelle  $X \rightarrow Y$  est complète si l'élimination d'un attribut  $x_i$  quelconque de  $X$  détruit la dépendance fonctionnelle.

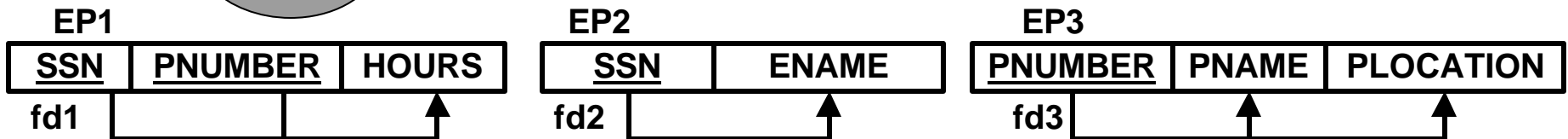
# 2NF: Deuxième forme normale

## Exemple:



Cette exemple n'est pas en 2NF, car certains attributs sont en dépendances fonctionnelles partielles.

NORMALISATION  
2NF



# 3NF: Troisième forme normale

---

R est une relation **3NF** ssi :

1. *elle est **2NF**,*
2. *tous les attributs “ non-clés ” sont dépendants non transitivement (dépendants directement) de la clé primaire.*

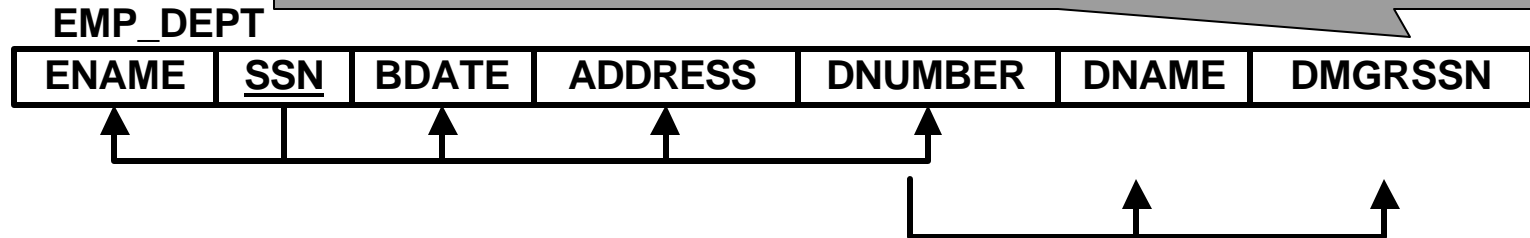
La **3NF** est basée sur le concept de dépendance fonctionnelle *transitive*.

Si les dépendances fonctionnelles  $X \rightarrow Z$  et  $Z \rightarrow Y$  existent. Alors, la dépendance fonctionnelle  $X \rightarrow Y$  est dite *transitive*.

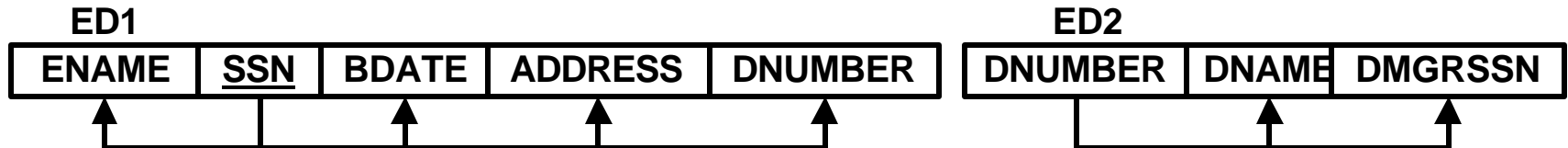
# 3NF: Troisième forme normale

## Exemple

DMGRSSN est une dépendance fonctionnelle transitive de SSN puisqu'on peut l'obtenir par  $SSN \rightarrow DNUMBER$  et  $DNUMBER \rightarrow DMGRSSN$



NORMALISATION  
3NF



# Processus de normalisation

---

Soit une relation en **1NF** : R

1. Obtenir les projections de R (1NF) (instanciation de la relation) pour faire l'analyse des données de la relation et **éliminer toute dépendance fonctionnelle partielle**.  
i.e. les attributs non clés doivent tous être complètement dépendants d'une clé primaire.
2. Obtenir les projections de R (2NF) pour faire l'analyse des données de la relation et **éliminer toute dépendance fonctionnelle transitive**.

# Conséquences du non respect de la normalisation

---

- *redondance de l'information; les attributs qui réfèrent à un attribut non clé seront répétés pour chaque apparition de cet attribut non clé.*
- *inconsistance; à toute les fois qu'une donnée est modifiée, toutes les répliques de ces données doivent être modifiées.*

# Avantages de la normalisation

---

## 1. Redondance réduite

- *moins de données dans la base, donc une plus petite BD,*
- *plus petite est la BD et plus rapides sont les E/S.*

## 2. Plus petites relations et plus petits tuples

- *plus de tuples peuvent être affichés ou imprimer sur une page,*
- *plus de tuples peuvent être traités en une même E/S,*
- *plus de tuples peuvent être mis en mémoire cache*

## 3. Assure que la BD est dans un état consistant

# Inconvénients de la normalisation

---

Par son nombre élevé de relations de petites tailles:

- ***n'optimise pas la performance** s'il faut effectuer plusieurs jonctions pour retrouver l'information.*
- *les jonctions peuvent être **coûteuses en temps CPU et en E/S.***

# La dénormalisation

---

## La dénormalisation...

- *est un processus intentionnel visant à s'éloigner de la normalisation pour améliorer les performances de la BD,*
- *peut être appliquée sur des colonnes ou des relations entières,*
- *s'effectue **après** la normalisation,*
- *requiert du concepteur de la BD de connaître comment les données seront utilisées,*

Elle peut être utilisé pour ajuster la BD à une application particulière.

# La dénormalisation

---

## Bénéfices

- *minimise le besoin de jonctions,*
- *réduit le nombre de clés étrangères (foreign key),*
- *réduit le nombre d'index,*
- *réduit le nombre de relations.*

**!** Le choix d'un bon schéma sera toujours un compromis entre la performance de la BD (moins de jonction) et l'absence d'anomalies de mise à jour (pas de redondance de données).

- *Le concepteur doit faire une étude sur la fréquence des besoins de certaines données pour justifier la redondance des données.*

# Exemple de conception d'un schéma relationnel

---

À partir du schéma E-R de l'exemple du chapitre 2, nous allons créer le schéma relationnel de la compagnie.

Voici les étapes:

1. Les **types d'entités normaux** deviennent une relation avec les mêmes attributs.

EMPLOYÉ(NAS, Prénom, Initial, NomFamille, Adresse, Salaire, Sexe, DateNaissance)

DÉPARTEMENT(Numéro, Nom, Local)

PROJET(Numéro, Nom, Local)

2. Les **types d'entités faibles** subissent la même transformation que les entités normales, mais en plus, on ajoute une clé étrangère (clé primaire du type d'entité forte correspondant).

DÉPENDANT(NAS, Nom, Sexe, DateNaissance, Relation)

# Exemple de conception d'un schéma relationnel

---

## 3. Pour la **relations 1 vers 1** (DIRIGE):

- *Soit inclure la clé primaire (NAS) d'EMPLOYÉ comme étant une clé étrangère dans DÉPARTEMENT*
- *Soit de combiner les deux entités.*

*Comme, il existe également une relation 1 vers N avec ces deux entités, nous allons retenir le premier choix.*

DÉPARTEMENT(Numéro, Nom, Local, NASDirecteur,  
DateDébutDirecteur)

# Exemple de conception d'un schéma relationnel

---

4. Pour les **relations 1 vers n**, il faut inclure la clé primaire de l'entité de cardinalité 1 comme étant une clé étrangère dans l'entité de cardinalité n.

◆ Pour la relation SUPERVISE, on ajoute un attribut qui s'appelle Superviseur.

EMPLOYÉ(NAS, Prénom, Initial, NomFamille, Adresse, Salaire, Sexe, DateNaissance, Superviseur)

◆ Pour la relation TRAVAILLE\_POUR, on ajoute le numéro de département dans employé.

EMPLOYÉ(NAS, Prénom, Initial, NomFamille, Adresse, Salaire, Sexe, DateNaissance, Superviseur, No\_Département)

◆ Pour la relation CONTRÔLE, on ajoute le numéro de département dans la relation PROJET.

PROJET(Numéro, Nom, Local, No\_Département)

# Exemple de conception d'un schéma relationnel

---

5. Pour **les relations n vers n**, il faut créer le tableau de la relation et inclure la clé primaire des entités dans ce tableau. La relation TRAVAILLE\_SUR devient :

TRAVAILLE\_SUR(NAS, No\_Projet, Heures)

Le résultat final de la transformation du modèle E-R en modèle relationnel est :

EMPLOYÉ(NAS, Prénom, Initial, NomFamille, Adresse, Salaire, Sexe, DateNaissance, Superviseur, No\_Département)

DÉPENDANT(NAS, Nom, Sexe, DateNaissance, Relation)

DÉPARTEMENT(Numéro, Nom, Local, NASDirecteur, DateDébutDirecteur)

PROJET(Numéro, Nom, Local, No\_Département)

TRAVAILLE\_SUR(NAS, No\_Projet, Heures)

# Exemple de conception d'un schéma relationnel

---

Pour terminer, il faut normaliser.

Ici, le modèle est simple, il n'y a pas qu'une modification à faire.

Pour la relation DÉPARTEMENT, puisqu'il peut y avoir plusieurs locaux (attribut valeurs multiples), il faut normaliser. Nous aurons ainsi :

DÉPARTEMENT(Numéro, Nom, NASDirecteur,  
DateDébutDirecteur)

DÉPARTEMENT\_LOCALISATION(No\_Département, local)

# Exemple de conception d'un schéma relationnel

La figure suivante présente le résultat de la normalisation 3NF de la BD COMPANY

Faire le schéma relationnel

...

**EMPLOYEE**

ENAME	<u>SSN</u>	BDATE	ADDRESS	DNUMBER
Smith, John B.	123456789	09-JAN-55	731 Fondren, Houston, TX	5
Wong, Franklin T.	333445555	08-DEC-45	638 Voss, Houston, TX	5
Zelaya, Alicia J.	999887777	19-JUL-58	3321 Castle, Spring, TX	4
Wallace, Jennifer S.	987654321	20-JUN-31	291 Berry, Bellaire, TX	4
Narayan, Remesh K.	666884444	15-SEP-52	975 Fire Oak, Humble, TX	5
English, Joyce A.	453453453	31-JUL-62	5631 Rice, Houston, TX	5
Jabbar, Ahmad V.	987987987	29-MAR-59	980 Dallas, Houston, TX	4
Borg, James E.	888665555	10-NOV-27	450 Stone, Houston, TX	1

**DEPARTMENT**

DNAME	<u>DNUMBER</u>	DMGRSSN
Research	5	333445555
Administration	4	987654321
Headquarters	1	888665555

**DEPT\_LOCATIONS**

<u>DNUMBER</u>	<u>DLOCATION</u>
1	Houston
4	Stafford
5	Bellaire
5	Sugarland
5	Houston

**WORKS\_ON**

<u>SSN</u>	<u>PNUMBER</u>	HOURS
123456789	1	32.5
123456789	2	7.5
666884444	3	40.0
453453453	1	20.0
453453453	2	20.0
333445555	2	10.0
333445555	3	10.0
333445555	10	10.0
333445555	20	10.0
999887777	30	30.0
999887777	10	10.0
987987987	10	35.0
987987987	30	5.0
987654321	30	20.0
987654321	20	15.0
888665555	20	null

**PROJECT**

PNAME	<u>PNUMBER</u>	PLOCATION	DNUM
ProductX	1	Bellaire	5
ProductY	2	Sugarland	5
ProductZ	3	Houston	5
Computerization	10	Stafford	4
Reorganization	20	Houston	1
Newbenefits	30	Stafford	4

NOTE: Il manque la table des dépendants