

Principes Des Bases De Données



Présenté par Gabriel MOPOLO MOKE, PAST UNS

Plan de l'enseignement



- Introduction
- Le modèle relationnel de CODD
- Pratique de l'algèbre relationnelle
- Structured English as a QUery Language
 - Pratique de SQL avec Oracle
 - Introduction à l'architecture du SGBD Oracle

Plan de l'enseignement



- La conception d'un schéma relationnel
 - Les liens
 - Comment obtenir le schéma relationnel ?
 - Les Formes Normales
 - Exemple d'un processus d'obtention d'un schéma relationnel

Principes Des Bases De Données



Introduction

Au cœur des SI !



- Les bases de données sont au cœur des Systèmes d'Informations (SI) des entreprises
- Elles résident partout
 - Du mainframe au PDA
 - Et même les cartes à puce !

BD ?!



- Une Base de Données (BD) ?
 - "Une **grande quantité de données**, centralisées ou non, servant aux besoins d'une ou de **plusieurs applications**, interrogeables et modifiables par un **groupe d'utilisateurs** travaillant en **parallèle**"

SGBD ?!



- Système de Gestion de Base(s) de Données (SGBD) ?
 - Le logiciel pour
 - Décrire, modifier, interroger et administrer la (les) BD
 - Bref,
 - "L'interface entre la base de données et les utilisateurs ou leurs programmes"
 - Un SGBD gère une ou plusieurs BD

Les plus des BD



- Certes, l'information est finalement stockée dans des fichiers, mais
 - Un système de fichiers, c'est
 - "Record at a time"
 - Une BD, c'est
 - "Set at a time"
- Et cela change tout !
 - Confort de manipulation

Les plus des BD




- Les BD/SGBD apportent une réelle Valeur Ajoutée par rapport aux fichiers
 - Manipulations des données par des non informaticiens !
 - Indépendance logique des données (vues différentes)
 - Caractère "générique" des données
 - Optimisation de l'accès à l'information

Les plus des BD



- Les BD/SGBD apportent une réelle Valeur Ajoutée par rapport aux fichiers (suite)
 - Indépendance du stockage physique
 - Partage des accès aux données
 - Concurrence d'accès
 - Sécurité des données
 - Sauvegarde et restauration

Architecture ANSI/SPARC d'un SGBD



□ Trois niveaux

□ Niveau interne

- Stockage physique des données

□ Niveau conceptuel


□ Structure et sémantique des données

- Indépendamment du niveau inférieur (i.e. implantation) ou supérieur (i.e. comment les utilisateurs s'en serviront)

□ Niveau externe


- Description pour chaque utilisateur de sa perception des données (couches d'abstraction)

Le background des SGBD d'aujourd'hui




- Les SGBD actuels disposent d'un background bien établi
 - En 1970, Ted CODD présente le **modèle relationnel**
 - En 1975, le prototype SYSTEM-R IBM en prouve la faisabilité
 - En 1977, Oracle V1
 - En 1984, IBM DB2
 - ...

Le background des SGBD d'aujourd'hui




- Les SGBD actuels disposent d'un background bien établi (suite)
 - Aujourd'hui Oracle, MS SQL Server, IBM DB2 sont "toujours" des SGBD **R** !
 - Dans lesquels l'objet est "plus ou moins" entré, pour donner des SGBD "Relationnel-Objet" aussi parfois appelé "Universels"

Le background des SGBD d'aujourd'hui



- Le modèle relationnel : la star des BD !
 - Il repose sur des concepts simples
 - Qui imposent de tout mettre à plat (simplicité et difficulté de la conception d'un schéma relationnel)
 - Mais puissant pour manipuler la BD
 - Accessible à des non informaticiens !
 - Approche ensembliste et non procédurale
 - On Line Transaction Processing (OLTP)

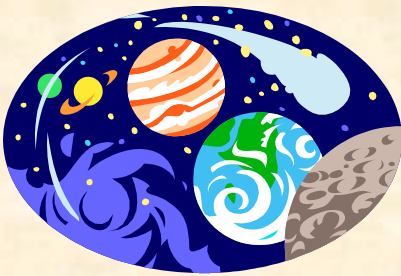
Le background des SGBD d'aujourd'hui



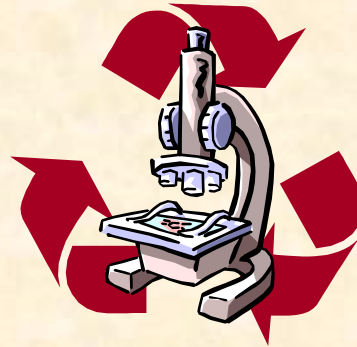
- Bien sûr le modèle relationnel a ses limites
 - Graphes complexes
 - SGBD Objets
 - Multimedia
 - SGBD dit "Universels" ou dédiés
- Analyse/Prévision/Simulation
 - Décisionnel : Decision Support System (DSS)
 - Analyse des données multidimensionnelles : On-Line Analytical Processing (OLAP), xOLAP

La modélisation d'une BD

- La modélisation des données permet d'identifier les entités de l'univers réel et les liens entre-elles



Univers réel



Modélisation

Modèle de données { Relationnel
Hiérarchique
Réseau

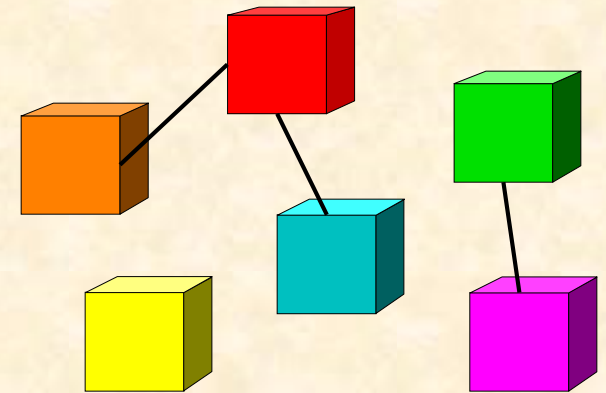


Schéma BD

Schéma de données { Relationnel
Hiérarchique
Réseau

La modélisation d'une BD



- Un modèle de données, c'est
 - Des structures
 - Des opérateurs
 - Des règles d'intégrité

Principes Des Bases De Données



**Le modèle relationnel
de CODD**

Le modèle de données relationnel de CODD



- Le modèle relationnel de CODD, c'est

- Structure

- Domaine, Relation, Attribut, Clé Primaire, Clé Étrangère

- Opérateurs

- Union, Intersection, Différence

- Sélection, Projection, Jointure, Division

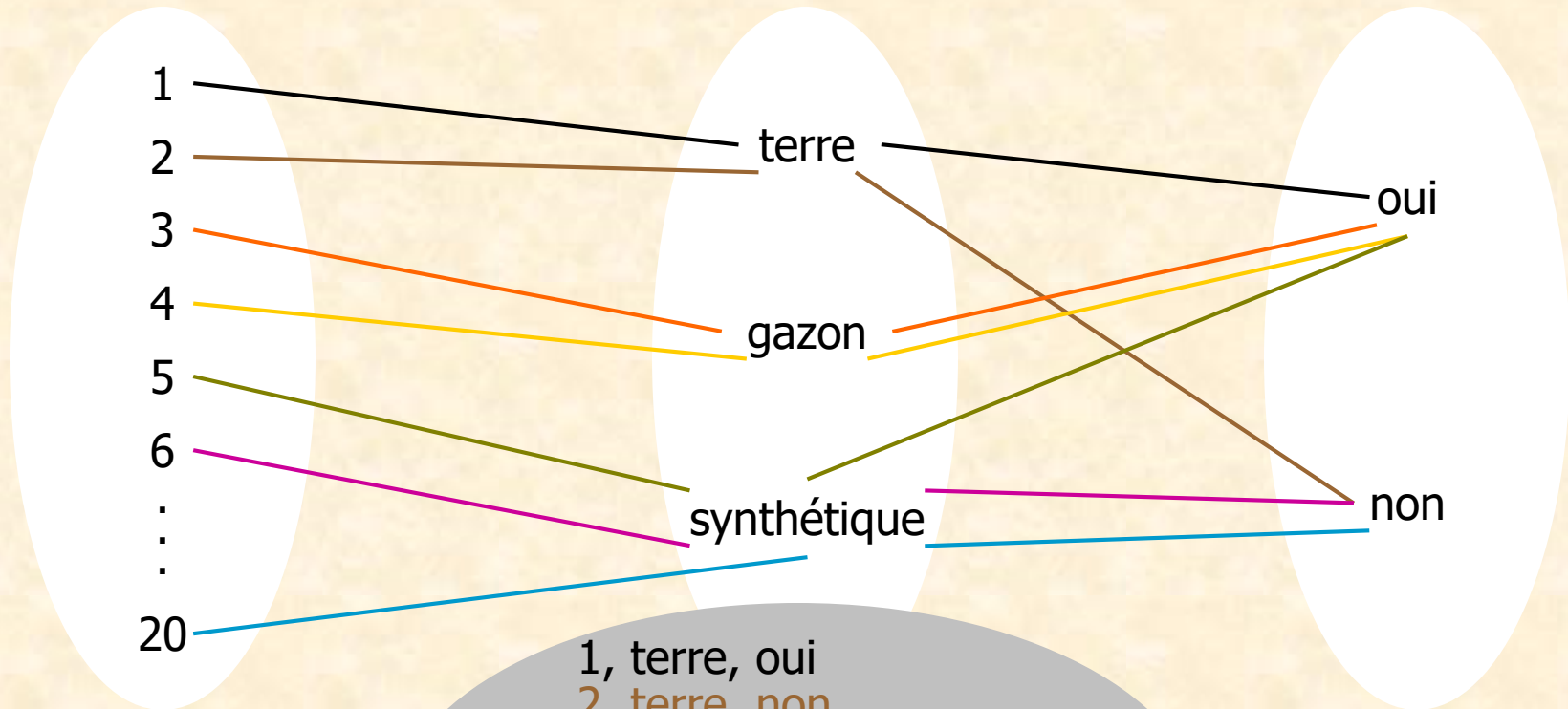
- Règles d'intégrité

- De domaine, de relation, de référence

Définition d'une Relation

- Sous-ensemble du produit cartésien de N ensembles
 - Chaque ensemble représente des valeurs possibles, e.g.
 - NUMERO={1..20}
 - TYPE={terre, gazon, synthétique}
 - NOM={1 à 30 lettres}
 - TELEPHONE={nul | 01..69999999}
 - Chaque ensemble représente un **domaine**
 - Le choix du nom d'un domaine est libre !

Définition d'une Relation



Relation COURT

1, terre, oui
2, terre, non
3, gazon, oui
4, gazon, oui
5, synthétique, oui
6, synthétique, non
...
20, synthétique, non

Définition d'une Relation

- Une relation est donc un ensemble de N -uplet (**tuples**)
 - La relation COURT se compose de triplets
 - La relation COURT est un sous-ensemble du produit cartésien des **domaines NUMERO, TYPE et DOUBLE**
 - Pour cette relation
 - Un élément du domaine NUMERO **doit** se trouver dans **un ou zéro triplet**
 - Un élément des domaines TYPE et DOUBLE **peut** se retrouver dans **zéro, un ou plusieurs triplets**

Définition et usage d'une Relation



- On définit une relation
- On "peuple" la relation en y insérant des tuples
 - Que l'on pourra modifier, supprimer
- Pour chaque tuple, la valeur de ses "composants" doit appartenir à l'ensemble des valeurs définies par les domaines sur lesquels ils reposent

Définition d'une Relation

□ Notation

□ La relation est définie par un **prédicat**

□ Le court de numéro NUMERO dispose d'une surface TYPE et accepte ou pas des parties en DOUBLE

□ Notation abrégée

- **NOM_RELATION(NOM_DOMAINE_a, NOM_DOMAINE_b, ...)**
- Par exemple :
COURT(NUMERO, TYPE, DOUBLE)

□ Les noms

- Des relations
 - Des domaines
- } **Sont libres !**

Définition d'une Relation

□ Notation (suite)

□ La relation se compose de **propositions** vraies

□ Le court numéro **1** dispose d'une surface en **terre** et **accepte** des parties en double

- COURT(**1**, **terre**, **oui**)

□ Le court numéro **5** dispose d'une surface en **synthétique** et **n'accepte pas** des parties en double

- COURT(**5**, **synthétique**, **non**)

□ La valeur d'un "composant" d'un tuple doit vérifier l'appartenance à son domaine de définition !

- **Intégrité de domaine**

Définition d'une Relation

□ L'attribut

□ Supposons une relation **RESERVATION** de prédicat

□ "*La réservation de numéro NUMERO s'opère de telle HEURE à telle HEURE*"

□ Le domaine HEURE peut-il figurer deux fois dans la relation RESERVATION sans ambiguïté ?

Définition d'une Relation

□ L'attribut (suite)

□ L'attribut qualifie le domaine dans une relation !

□ "La réservation de numéro NUMERO s'opère de
HEURE_DEBUT à HEURE_FIN"

- RESERVATION(NUMERO, HEURE_DEBUT, HEURE_FIN)

□ Notation :

NOM_RELATION(NOM_ATTRIBUT₁, ..., NOM_ATTRIBUT_n)

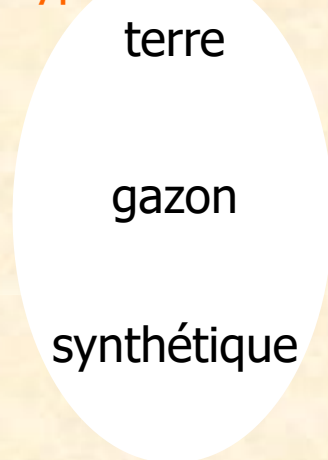
Définition d'une Relation

- L'attribut (suite)
 - À un instant t , un attribut possède **une seule valeur** α_t ou **aucune**
 - Il n'est PAS multi-valué (NF1...) !
 - **Intégrité de domaine** : $\alpha_t \in \text{domaine}$
 - Le domaine acceptant ou pas la valeur nulle

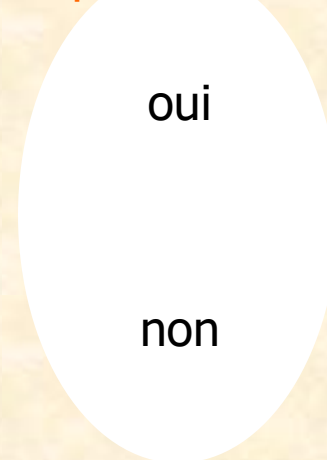
Le domaine : un concept sémantique

- Le domaine est un concept **sémantique**
 - Distingue 2 ensembles syntaxiq^t compatibles

Type de surface



Accepte les doubles ?



Syntaxique

$x \in \{\text{chaîne de caractères}\}$

$x \in \{\text{chaîne de caractères}\}$

Sémantique

$x \in \{\text{terre, gazon, synthétique}\}$

$x \in \{\text{oui, non}\}$

terre θ oui n'a pas de sens sémantique !

Le domaine : un DOUBLE concept sémantique

- De **définition** (intégrité de domaine)
 - $\alpha_t \in \text{domaine}$
- De **manipulation**
 - $\text{Domaine}_k \theta \text{Domaine}_k$
 - $\text{Domaine}_k \times \text{Domaine}_i$
- Le domaine joue un rôle essentiel !
 - Les SGBD le considèrent-ils pour autant ?

Clé primaire et domaine primaire

- Clé primaire d'une relation
 - Attribut OU un groupe d'attributs identifiant de façon unique chaque tuple de la relation
 - La relation étant un ensemble, tous ses éléments sont distincts !
 - Toute relation possède une CP unique et non nulle (intégrité de {relation|entité})
 - Elle peut posséder 1 ou N clé candidate
- Domaine primaire
 - Domaine de définition d'une CP **mono-formée**

Clé étrangère

□ Clé étrangère

- Attribut d'une relation défini sur un domaine primaire mais qui n'est pas CP de cette relation
- Sert à tisser les liens entre les relations
 - La CÉ "*référence/pointe/désigne*" un tuple d'une [autre] relation
 - Qu'elle identifie grâce à la valeur de sa CP
intégrité de référence

Clé primaire et étrangère

□ Conventions de notation

□ Une CP s'accompagne souvent d'un # ou est soulignée

□ COURT(NUMERO#, TYPE, DOUBLE)

□ COURT(NUMERO, TYPE, DOUBLE)

□ Une CÉ s'accompagne souvent de ## ou est soulignée en pointillés

□ RESERVATION(NUMERO#, COURT##, ...)

□ RESERVATION(NUMERO, COURT, ...)

Un schéma relationnel

□ Définition des domaines

□ **NOM_DU_DOMAINE**={ensemble des valeurs}

□ Le choix du nom du domaine est libre !

□ Un domaine définit les valeurs possibles des attributs définis sur ce domaine

□ C'est à la création ou modification du tuple que le ou les attribut(s) reçoivent leur valeur

□ **À un instant t**

- $\alpha_t \in \text{domaine}$
- α_t ne vaut qu'une seule valeur ou aucune

Un schéma relationnel

□ Définition des domaines (suite)

□ Exemple

- NUMERO_COURT={1..20} primaire
- TYPE_COURT={terre, gazon, synthétique}
- DOUBLE={oui, non}
- NUMERO_RESERVATION={1..9999} primaire
- JOURS_OUVRES={jours ouverts du club}
- HEURE={8..20}
- NB_JOUEURS={2,4}

Un schéma relationnel

□ Définition des relations

□ $NOM_RELATION(NOM_ATTRIBUT_1, \dots)$

□ Chaque attribut est défini sur un domaine

- $NOM_ATTRIBUT$ défini sur $NOM_DOMAINE$

□ Exemple

□ $COURT(NUMERO\#, TYPE, DOUBLE)$

- $NUMERO\#$ défini sur $NUMERO_COURT$
- $TYPE$ défini sur $TYPE_COURT$
- $DOUBLE$ défini sur $DOUBLE$

Un schéma relationnel

□ Définition des relations (suite)

□ Exemple (suite)

□ RESERVATION(NUMERO#, COURT##, DATE, HEURE_DEBUT, HEURE_FIN, NB_JOUEURS)

Intégrité
de
référence !

- NUMERO# défini sur NUMERO_RESERVATION
- COURT## défini sur NUMERO_COURT
 - CÉ vers COURT.NUMERO#
- DATE défini sur JOURS_OUVRES
- HEURE_DEBUT défini sur HEURE
- HEURE_FIN défini sur HEURE
- NB_JOUEURS défini sur NB_JOUEURS

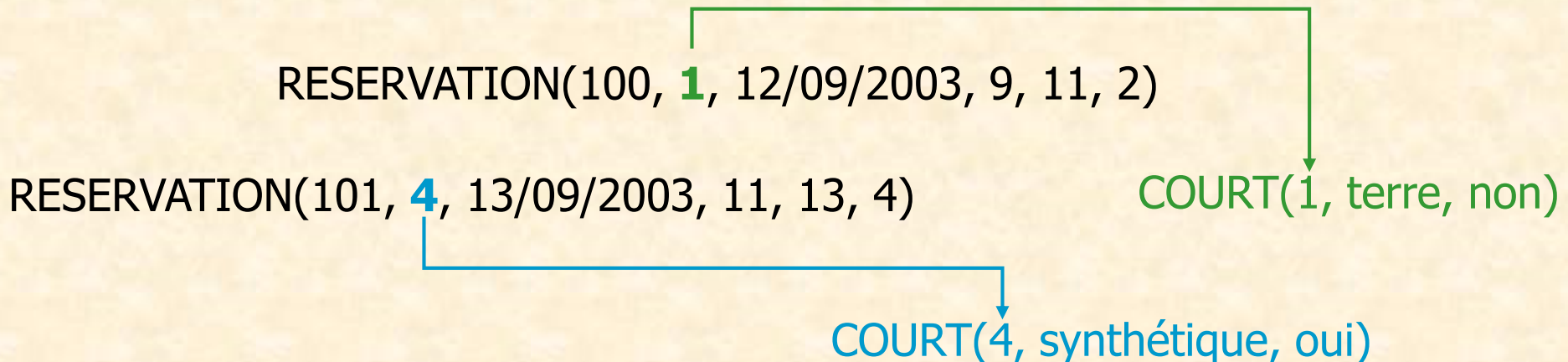
Un schéma relationnel

□ Définition des relations (suite)

□ Exemple (suite)

□ RESERVATION(NUMERO#, COURT##, DATE, HEURE_DEBUT, HEURE_FIN, NB_JOUEURS)

- La CÉ COURT## "référence/pointe/identifie" un tuple de la relation COURT



Un schéma relationnel



□ Règles de gestion

- "Le club est ouvert de 8H à 20H"

- "Seul un membre peut réserver un court"

□ Comment les prendre en compte ?

Un schéma relationnel



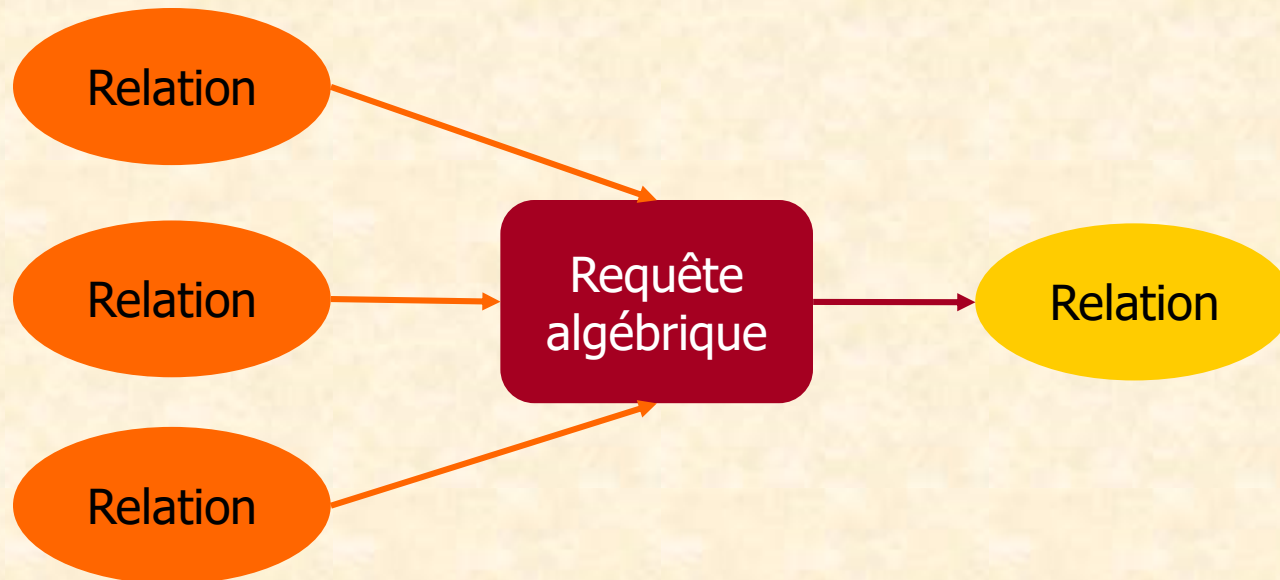
□ Règles de gestion

- "Il ne peut pas y avoir deux membres de même nom, prénom et téléphone"
- "Un court de type simple ne peut pas être réservé pour un double"
- "Un court ne peut pas être réservé le même jour à la même heure ou à des heures se chevauchant"
- "Un membre ne peut pas réserver plus d'un court par jour"

□ Comment les prendre en compte ?

Le langage algébrique

- L'interrogation d'une ou de plusieurs relations s'opère par l'application d'opérateurs algébriques
 - Exprimés dans une requête algébrique



Le langage algébrique

- Une requête algébrique se déroule sur une ou plusieurs lignes
 - Chaque ligne traduit l'application d'un opérateur algébrique sur
 - Une relation "de départ"
 - Une relation "intermédiaire"
 - Obtenue par une ligne précédente

Le langage algébrique

□ Rappel

- Relation = ensemble, pas de doublons !

□ Les opérateurs algébriques

□ Ensemblistes

- Union, Intersection, Différence, Produit Cartésien

□ Restrictifs

- Unaires : Projection, Sélection

- Binaire : Division

□ Extensif binaire : Jointure

Le langage algébrique

- Les opérateurs algébriques ensemblistes
 - Ils s'appliquent sur des ensembles **uncompatibles**, i.e. sur des relations disposant
 - Du même nombre d'attributs
 - Par rang définis sur le même domaine
- L'opérateur **Union**
 - $R_1 \cup R_2$ unie les tuples de R_2 à R_1 et élimine les doublons, puisque le résultat est une relation et donc un ensemble

Le langage algébrique

- Les opérateurs algébriques ensemblistes
 - L'opérateur **Intersection**
 - $R_1 \cap R_2$ réduit l'ensemble aux tuples communs à R_1 et R_2
 - L'opérateur **Différence**
 - $R_1 - R_2$ de R_1 retire R_2
 - "J'enlève ce que je NE veux PAS !"
 - L'opérateur **Produit Cartésien**
 - $R_1 \times R_2$

Le langage algébrique

□ Les opérateurs algébriques restrictifs unaires

□ L'opérateur **Projection**

COURT		
1,	terre,	oui
2,	terre,	non
3,	gazon,	oui
4,	gazon,	oui
5,	synthétique,	non
6,	synthétique,	non
...		
20,	synthétique,	non

□ $R_i = \text{Projection COURT}(\text{NUMERO\#}, \text{TYPE})$

- De quel type de surface dispose chaque court ?

Le langage algébrique

□ Les opérateurs algébriques restrictifs unaires

□ L'opérateur Sélection

Est retenu chaque tuple satisfaisant à la condition

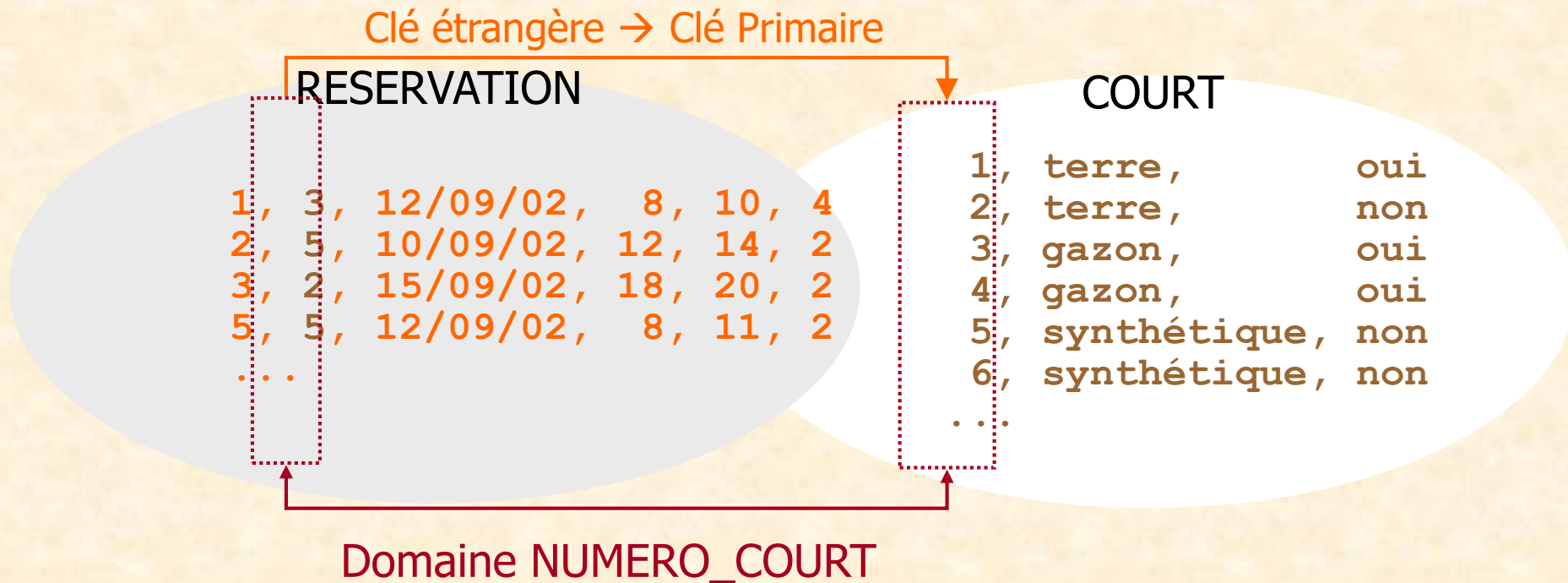
1,	terre,	oui
2,	terre,	non
3,	gazon,	oui
4,	gazon,	oui
5,	synthétique,	non
6,	synthétique,	non
...		
20,	synthétique,	non

□ R_i = Sélection COURT(TYPE=gazon)

- Quels sont les courts en gazon ?


Le langage algébrique

□ L'opérateur algébrique extensif binaire Jointure



Le langage algébrique

□ L'opérateur algébrique extensif binaire Jointure



1	3	12/09/02	8	10	4	3	gazon	oui
2	5	10/09/02	12	14	2	5	synthétique	non
3	2	15/09/02	18	20	2	2	terre	non
5	5	12/09/02	8	11	2	5	synthétique	non
.....								
						4	gazon	oui
.....								
						6	synthétique	non
.....								

- $R_i = \text{Jointure RESERVATION}(\text{COURT}\#\# = \text{NUMERO}\#)\text{COURT}$
 - Rappel : $\text{Domaine}_k \theta \text{Domaine}_k$

Le langage algébrique

□ L'opérateur algébrique extensif binaire

Jointure

□ La plus part du temps s'applique sur un lien
 $CE \rightarrow CP$

□ Reconstruction de l'information

□ Cas général : $\text{Domaine}_k \theta \text{Domaine}_k$!

□ Y compris auto-jointure ("sélection paramétrée")

Le langage algébrique

□ L'opérateur algébrique restrictif binaire

Division

□ Forme particulière $(\alpha, \beta) \div \beta$

GOURMAND(PERSONNE, DESSERT)

Emma	Baba au rhum
Emma	Chantilly
Emma	Tarte au citron
Emma	Tiramisu
Pierre	Baba au rhum
Pierre	Chantilly
Pierre	Tiramisu
Thomas	Chantilly
Thomas	Tiramisu



DESSERT(NOM)

Baba au rhum
Chantilly
Tarte au citron
Tiramisu

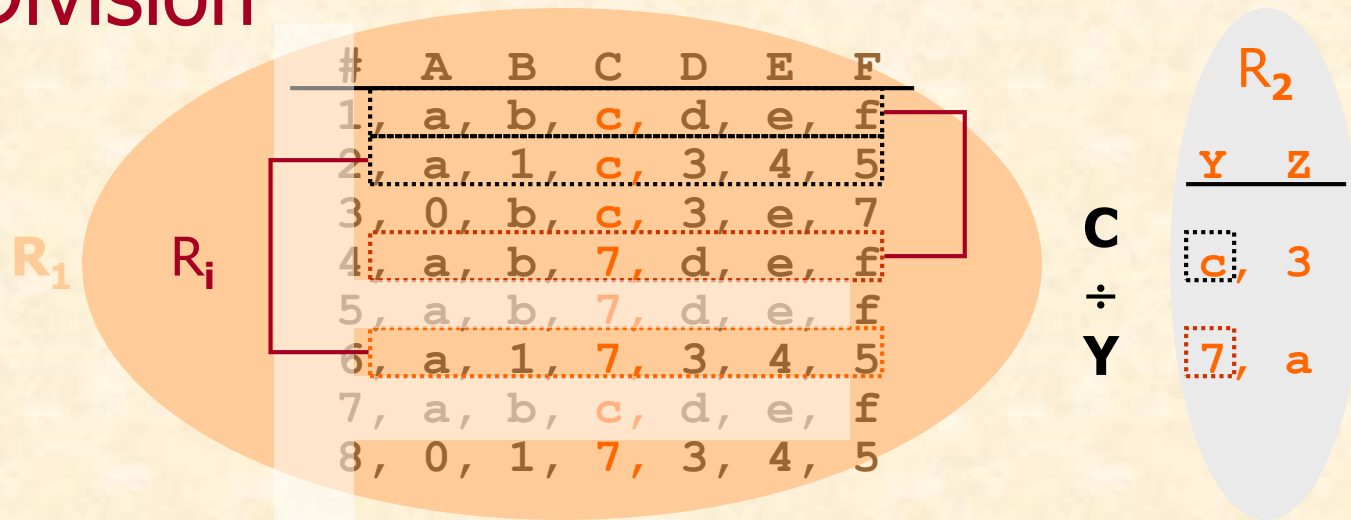


Emma

Le langage algébrique

□ L'opérateur algébrique restrictif binaire

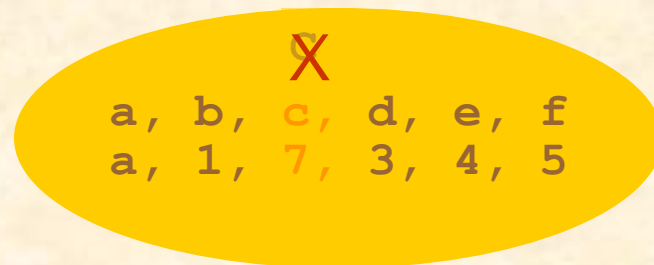
Division



? $R_i = \text{Projection } R_1(A, B, C, D, E, F)$

Division $R_i(C=Y) R_2$

"Quels sont les tuples de R_i dont l'attribut C contient c puis 7 ?"



Le langage algébrique

□ L'opérateur algébrique restrictif binaire

Division

□ Forme particulière : $(\alpha, \beta) \div \beta$

□ Forme générale

$R = \text{Division } R_1(\text{attribut}_i = \text{attribut}_k) R_2$

□ R contient les tuples $(\text{attribut}_1, \text{attribut}_2, \dots, \text{attribut}_n$; sauf attribut_i) de R_1 pour lesquels ses répétitions consomment pour l'attribut_i toutes les valeurs λ de attribut_k de R_2

□ Rappel : $\text{Domaine}_k \theta \text{Domaine}_k$

Le langage algébrique

□ L'opérateur algébrique restrictif binaire

Division

□ Ne pas confondre la division avec la sélection !

□ $R = \text{Sélection } R_1(\text{attribut}_i=a \text{ Et attribut}_i=b)$

- N'a pas de sens !

Principes Des Bases De Données



**Pratique de l'algèbre
relationnelle**

Soit le schéma relationnel suivant



→ Lien Clé Étrangère vers Clé Primaire

Soit le schéma relationnel suivant

□ Les domaines

- NUMERO_COURT={1..20} primaire
- TYPE_COURT={terre, gazon, synthétique}
- DOUBLE={oui, non}
- NUMERO_RESERVATION={1..9999} primaire
- HEURE={8..20}
- JOURS_OUVRES={jours ouverts du club}
- NB_JOUEURS={2,4}

Soit le schéma relationnel suivant

□ Les domaines

- NUMERO_MEMBRE={1..600} primaire
- NOM={1 à 30 lettres}
- PRENOM={1 à 20 lettres}
- TELEPHONE={nul | 01..69999999}
- VILLE={1 à 30 lettres}
- AGE={5..99}
- SEXE={h,f}
- CLASSEMENT={nul | 0,0..30,0}

Soit le schéma relationnel suivant

□ Les relations

MEMBRE (NUMERO#, NOM, PRENOM, CLASSEMENT, AGE, SEXE, VILLE, TELEPHONE)

NUMERO#	défini sur NUMERO_MEMBRE	Clé Primaire
NOM	défini sur NOM	
PRENOM	défini sur PRENOM	
CLASSEMENT	défini sur CLASSEMENT	
AGE	défini sur AGE	
SEXE	défini sur SEXE	
VILLE	défini sur VILLE	
TELEPHONE	défini sur TELEPHONE	

COURT (NUMERO#, TYPE, DOUBLE)

NUMERO#	défini sur NUMERO_COURT	Clé Primaire
TYPE	défini sur TYPE_COURT	
DOUBLE	défini sur DOUBLE	

Soit le schéma relationnel suivant

□ Les relations

RESERVATION (NUMERO#, COURT##, MEMBRE##, DATE,
HEURE_DEBUT, HEURE_FIN, NB_JOUEURS)

NUMERO#	défini sur	NUMERO_RESERVATION	Clé Primaire
COURT##	défini sur	NUMERO_COURT	CÉ->COURT.NUMERO#
MEMBRE##	défini sur	NUMERO_MEMBRE	CÉ->MEMBRE.NUMERO#
DATE	défini sur	CALENDRIER	
HEURE_DEBUT	défini sur	HEURE	
HEURE_FIN	défini sur	HEURE	
NB_JOUEURS	défini sur	NB_JOUEURS	

Pratique de l'algèbre relationnelle



- En utilisant l'algèbre relationnelle, répondez aux questions suivantes
 - Cf. le tiré à part
 - Les conditions seront exprimées **sans et/ou**, pour utiliser plus souvent les opérateurs algébriques...

Principes des bases de données



Les commandements du relationnel

Les commandements du relationnel




- Un SGBDR est capable de gérer entièrement la ou les BD avec des fonctions relationnelles
- Règle d'information
 - Une BDR représente au niveau logique les informations uniquement sous la forme relations/attributs

Les commandements du relationnel



- Règle d'accès garanti
 - Toute valeur d'attribut est accessible via le nom de la relation, celui de l'attribut et la valeur de la clé primaire
- Les valeurs nulles
 - Elles sont prises en compte dans une BDR
- Le dictionnaire des données
 - Il est représenté au niveau logique sous la forme relations/attributs

Les commandements du relationnel



□ LDD et LMD

- Un SGBDR admet un Langage de Définition Des Données et un Langage de Manipulation Des Données

- Et un Langage de Contrôle des Données

□ Mise à jour ensemblistes

- Une relation ou le résultat d'une requête peut être un argument de recherche ou de mise à jour

Les commandements du relationnel




- Indépendance Physique
 - Transparence des méthodes d'accès (e.g. index), des méthodes de stockage et du support physique et de la localisation des données
- Contraintes d'intégrité
 - Les contraintes d'intégrité de l'univers réel modélisé doivent pouvoir être représentées

Principes Des Bases De Données




SQL

SQL



- SQL, Initialement Structured English as a QUery Langage - IBM System-R 1975
 - Est un Langage de
 - Définition des Données (LDD)
 - Manipulation des Données (LMD)
 - Contrôle des données (LCD)
 - Pour les SGBDR reconnu par l'ANSI et l'ISO
 - SQL/86 (SQL 1) en 1986
 - SQL/92 (SQL 2) en 1989
 - SQL/99 (SQL 3) en 1999 Orienté Objet
 - <http://www.sqlstandards.org>


SQL



- SQL-92 définissait quatre niveaux de conformité
 - *Entry, Transitional, Intermediate* et *Full*
 - Une implantation de SQL \geq *Entry*


- SQL-99
 - Conformance minimale : *Core*
 - Sur-ensemble de SQL-99 *Entry*

SQL



- SQL parle de
 - Table
 - Colonne
 - Ligne
 - Le modèle relationnel définit
 - Relation
 - Attribut
 - Tuple
 - et Domaine !
- (SQL 2)


SQL



- SQL n'est pas sensible à la casse*
- SQL peut être utilisé depuis
 - Un outil de requêtage textuel ou graphique
 - Depuis un programme (embed SQL) ou par un programme via une API (e.g. ODBC, JDBC)
- Les commentaires en SQL (non imbriqués)
 - -- une ligne
 - /* */ N lignes

* Sur les mots clés et noms d'objets, pas sur les valeurs caractères !

SQL



- Les chaînes de caractères s'expriment entre **simple** quotes (i.e. '...')
- Dans un requêteur, une commande SQL se termine par **;**

Principes Des Bases De Données



SQL - LMD

Les commandes du LMD



□ Select

- Interrogation

□ Insert

- Insertion

□ Update

- Mise à jour

□ Delete

- Suppression

Principes Des Bases De Données



SQL – LMD
L'interrogation

L'interrogation

- S'opère par l'instruction **Select**
 - Ne pas réduire à l'op. algébrique **Selection** !!!

Select NOM_COLONNE, fonction, ...

Projection

From NOM_TABLE, ...

Where conditions

Sélection, Jointure, Division

Group By

partitionnement

Having

critères sur le partitionnement

Order By

Union, Minus, Intersect

Union, Différence, Intersection

Select ...

L'interrogation, des exemples de Select

```
Select NOM, PRENOM  
From MEMBRE;
```

```
Select NOM, PRENOM, AGE  
From MEMBRE  
Where AGE > 25 And VILLE='Nice';
```

```
Select Min(AGE), Max(AGE)  
From MEMBRE  
Where VILLE='Nice' And Sexe='f';
```

```
Select *  
From MEMBRE;
```

```
Select *  
From COURT  
Where TYPE='terre' And DOUBLE='oui';
```

```
Select NOM, PRENOM  
From MEMBRE  
Where AGE > 25  
Intersect  
Select NOM, PRENOM  
From MEMBRE  
Where VILLE='Nice';
```

L'instruction Select

□ Syntaxe générale

```
SELECT [ALL | DISTINCT] {[schéma.table].* |  
    [expr]} [c_alias] [constante], ...  
FROM    [schéma].élément [t_alias], ...  
[WHERE <condition>]  
[CONNECT BY <condition>  
    [START WITH <condition>]]  
[GROUP BY expr, ...  
    [HAVING <condition>]]  
[ORDER BY {expr|col} [ASC|DESC], ...
```

Principes Des Bases De Données



SQL – LMD

L'interrogation

Implantation de la Projection

L'instruction Select

□ Implantation de la Projection

Select critère de projection[, cdp, cdp, ...]

From TABLE

□ Les critères de projection pouvant être

- Colonnes, *
- Calculs horizontaux ou verticaux (expression)
- Constante
- Renommage de colonne ou d'expression

L'instruction Select

□ Implantation de la Projection

□ Critères de projection = expression

□ Calculs horizontaux

- `Select NOM, PRENOM, SALAIRE*12
From EMPLOYE;`

□ Calculs verticaux (e.g. Count, Avg, Max, Min, Sum)

- `Select Count(*)
From EMPLOYE;`
- `Select Sum(SALAIRE)
From EMPLOYE;`
- `Select Min(SALAIRE), Max(SALAIRE), Avg(SALAIRE)
From EMPLOYE;`

L'instruction Select



- Implantation de la Projection
 - Critères de projection = constante
 - Valeur fixe exprimée entre apostrophes)
 - Select NUMERO, 'est un court en' COURT_EN, TYPE
From COURT;

L'instruction Select



- Implantation de la Projection
 - Critères de projections = renommage de colonne ou expression
 - `Select NOM, PRENOM, SALAIRE*12 "Salaire annuel"`
`From EMPLOYE;`
 - `Select Sum(SALAIRE) "La masse salariale mensuelle"`
`From EMPLOYE;`

L'instruction Select

- Implantation de la Projection
 - Par défaut les tuples doublons sont conservés !
 - Élimination par le mot clé **Distinct**
 - `Select Distinct cdp[, cdp, cdp, ...]`
From

Question :

Select Count(TYPE)
From COURT;

Select Count(Distinct TYPE)
From COURT;

Select Distinct Count(TYPE)
From COURT;

Principes Des Bases De Données



SQL – LMD

L'interrogation

Implantation de la Sélection

L'instruction Select

- Implantation de la Sélection

Select cdp[, cdp, cdp, ...]

From TABLE

Where [NOT] condition [AND | OR]...

- Condition étant de la forme

opérande opérateur opérande, avec

- **opérande** \in {colonne|expr|constante}

ou même une requête pour le deuxième opérande !

- **opérateur** \in {=, !=, <=, <, >, >=, In, Like}

L'instruction Select



□ Implantation de la Sélection

- Select NOM, PRENOM
From MEMBRE
Where NOM = 'MARTIN';

- Select NOM, PRENOM
From MEMBRE
Where NOM = 'MARTIN' And PRENOM='Pierre';

- Select *
From MEMBRE
Where AGE Between 20 And 50;

L'instruction Select

□ Implantation de la Sélection

□ Chaînes de caractères, l'opérateur like

□ _ représente un caractère

□ % représente N caractères

□ Select NOM, PRENOM

From MEMBRE

Where NOM Like 'M _ _ _ _';

□ Select NOM, PRENOM

From MEMBRE

Where NOM Like 'M%';

L'instruction Select

□ Implantation de la Sélection

□ Is Null et Is Not Null

□ Select NOM, PRENOM
From MEMBRE
Where CLASSEMENT Is Null;

□ Select NOM, PRENOM
From MEMBRE
Where CLASSEMENT Is Not Null;

□ Ne pas utiliser = Null ni != Null

□ No rows selected !

L'instruction Select

□ Implantation de la Sélection

□ Correspondance avec un ensemble

□ Appartenance : In (ou \notin : Not In)

- Select *
From COURT
Where TYPE In ('terre', 'gazon');

□ Comparaison

- Where {col,expr}{=, !=, >, >=, <, <= }[Any,Some,All] (...)
- Any \Leftrightarrow Some, =Any \Leftrightarrow In, !=All \Leftrightarrow Not In
- Utilité ?!
 - Where AGE <All (25,42,23,35,28,50,49)

Principes Des Bases De Données



SQL – LMD

L'interrogation

Requêtes imbriquées

INdépendantes

Requêtes imbriquées indépendantes

□ Correspondance avec un ensemble

□ Select cdp[, cdp, ...]

From

Where {col,expr} θ (Select ...)

□ Avec

□ Si (Select ...) ramène pas plus d'une ligne

$\theta \in \{=, !=, >, >=, <, <=, \text{In}, \tau \text{ Any}, \tau \text{ Some}, \tau \text{ All}\}$

□ Sinon

$\theta \in \{\text{In}, \tau \text{ Any}, \tau \text{ Some}, \tau \text{ All}\}$

Requêtes imbriquées indépendantes

□ Correspondance avec un ensemble

```
Select NOM, PRENOM
From MEMBRE
Where VILLE = ( Select VILLE
                  From MEMBRE
                  Where NOM='HENRY' And PRENOM='EMMA' )
      And ( NOM!='HENRY' Or PRENOM!='EMMA' );
-- Pas d'homonymes supposés
```

```
Select NOM, PRENOM
From MEMBRE
Where VILLE In ( Select VILLE
                  From MEMBRE
                  Where AGE=40 );
```

Requêtes imbriquées indépendantes

□ Correspondance avec un ensemble

```
Select NOM, PRENOM
From MEMBRE
Where AGE >= All( Select AGE
                  From MEMBRE
                  Where VILLE='Antibes' )
And VILLE!='Antibes' ;
```

```
Select NOM, PRENOM
From MEMBRE
Where AGE >= ( Select Max(AGE)
              From MEMBRE
              Where VILLE='Antibes' )
And VILLE!='Antibes' ;
```

Requêtes imbriquées indépendantes



- Correspondance avec un ensemble

```
Select NOM, PRENOM  
From MEMBRE  
Where NUMERO# Not In ( Select MEMBRE##  
                        From RESERVATION );
```

Principes Des Bases De Données



SQL – LMD

L'interrogation

Implantation de la jointure

Implantation de la jointure

```
Select NOM, PRENOM  
From MEMBRE  
Where NUMERO# In( Select MEMBRE##  
                  From RESERVATION );
```

- Jointure sous forme ensembliste !
 - Projection limitée aux colonnes de la relation "de départ"

Implantation de la jointure

□ Forme prédicative

□ Where [table.]colonne = [table.]colonne

□ Select NOM, PRENOM

From RESERVATION, MEMBRE

Where **MEMBRE##=MEMBRE.NUMERO#** ;

-- Pour équivalence avec la forme ensembliste

-- précédente : utiliser Distinct !

□ Préfixer les noms ambigus des colonnes par celui des tables

Implantation de la jointure

□ Forme prédicative

```
Select COURT.NUMERO#, TYPE, HEURE_DEBUT, HEURE_FIN, NOM, PRENOM  
From RESERVATION, MEMBRE, COURT  
Where LA_DATE='20/09/2003' And MEMBRE## = MEMBRE.NUMERO#  
      And COURT## = COURT.NUMERO#;
```

Implantation de la jointure

□ La jointure externe

- i.e. avoir quand même les lignes non référencées par une clé étrangère
- Where [relation.]cléPrimaire = [relation.]cléÉtrangère (+)

```
Select NOM, PRENOM, COURT##, LA_DATE
From RESERVATION, MEMBRE
Where MEMBRE.NUMERO# = MEMBRE## (+);
```

NOM	PRENOM	COURT##	LA_DATE
JOSSE	CHRISTELLE		
DURAND	PIERRE		
DUPONT	CHARLES	1	15/09/02
DUPONT	CHARLES	1	16/09/02
...			

Principes Des Bases De Données



SQL – LMD

L'interrogation

Implantation de l'union**,
intersection et **différence****

Implantation de l'union, intersection et différence

□ Opérateurs ensemblistes binaires

□ Union [All]

Select cdp[, cdp, cdp, ...]

□ Intersect

From

Where

□ Minus

{Union [All], Intersect, Minus}

Select cdp[, cdp, cdp, ...]

From

Where

□ Chaque Select doit retourner un ensemble unicompatible syntaxiquement !

Principes Des Bases De Données



SQL – LMD

L'interrogation

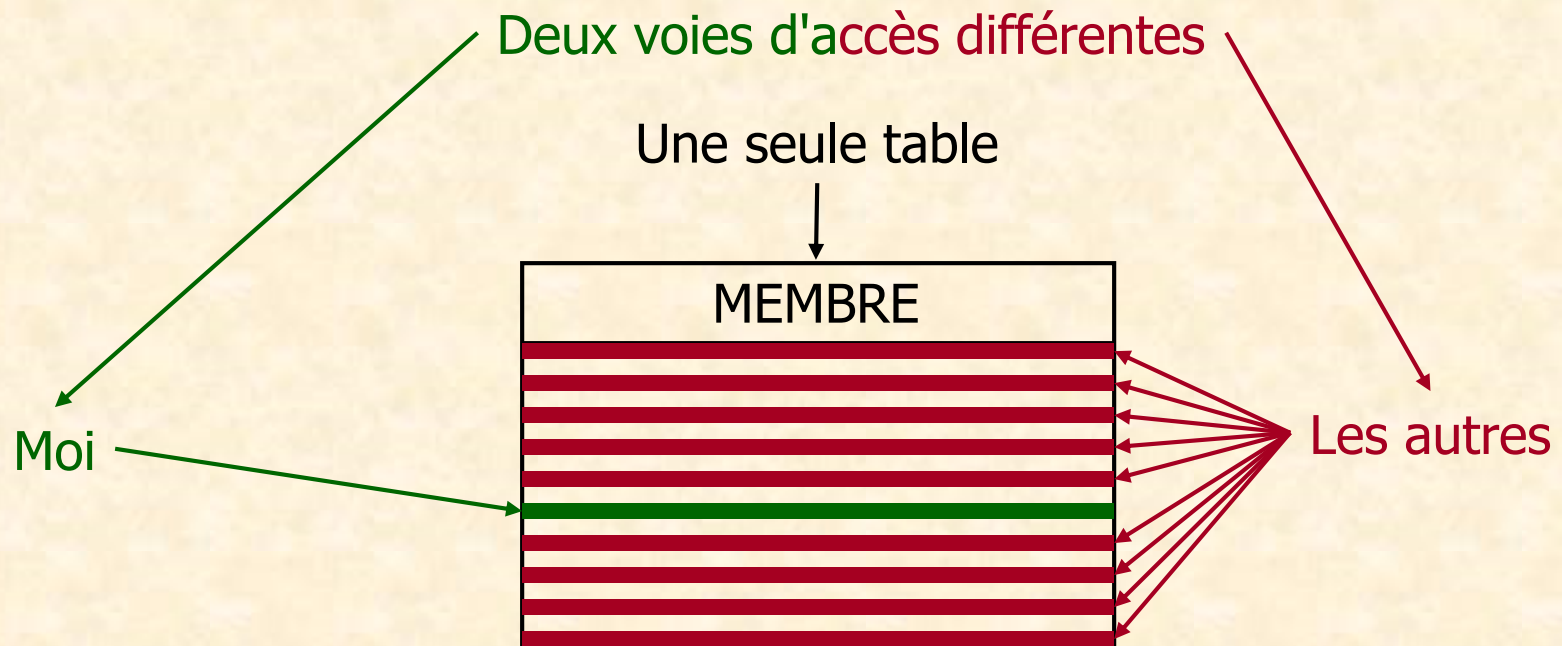
L'auto-jointure

ou

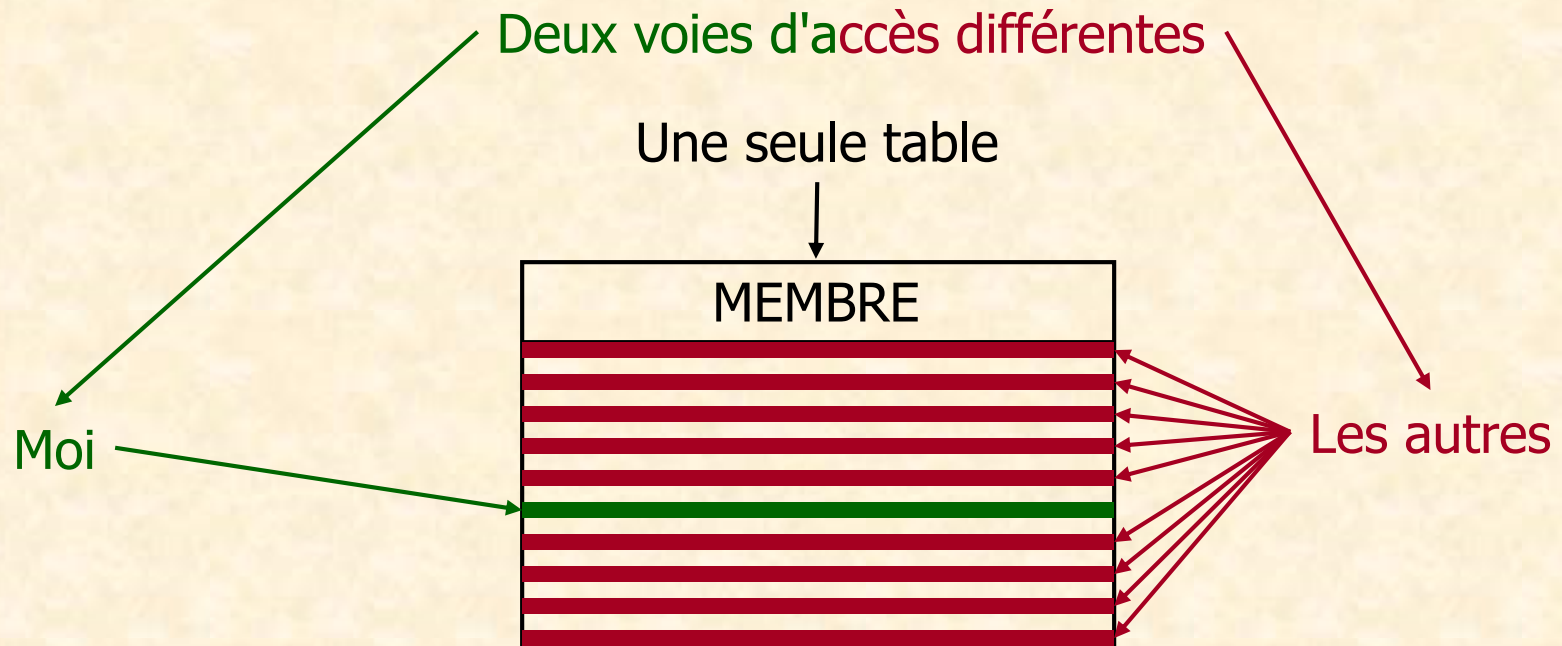
les variables de parcours

L'auto-jointure

- Besoin d'accéder par des voies différentes à la même table : **variable de parcours**
 - Qui habite dans la même ville que moi ?



L'auto-jointure



```
Select M1.VILLE, M1.NOM, M1.PRENOM, M2.NOM, M2.PRENOM
From MEMBRE M1, MEMBRE M2
Where M1.VILLE = M2.VILLE And M1.NUMERO# < M2.NUMERO#
```

Principes Des Bases De Données



SQL – LMD

L'interrogation

Requêtes imbriquées

DÉpendantes

Requêtes imbriquées DÉPENDANTES

□ Correspondance synchronisée avec un ensemble

□ Select

From NOM_TABLE T1

Where {col,expr} θ (

Select ...

From AUTRE_TABLE

Where T1.COL θ AUTRE_TABLE.COL)

Test d'une existence

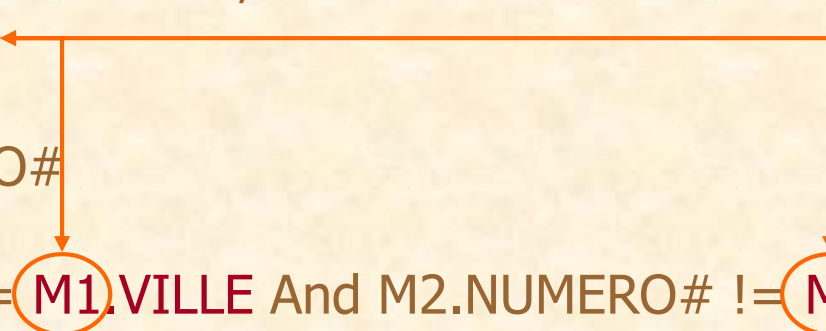


- Conditionner le résultat par rapport à une existence (ou une absence)
 - Select
From
Where [Not] Exists (Select ...)
- (Select ...) retourne vrai ou faux

Requêtes imbriquées DÉPENDANTES

- Correspondance synchronisée avec un ensemble
 - Quels sont les membres habitant seul dans une ville et de quelle ville s'agit-il ?

```
Select M1.NOM, M1.PRENOM, M1.VILLE
From MEMBRE M1
Where Not Exists (
  Select M2.NUMERO#
  From MEMBRE M2
  Where M2.VILLE = M1.VILLE And M2.NUMERO# != M1.NUMERO# );
```



Principes Des Bases De Données



SQL – LMD

L'interrogation

Implantation de la **division**

La Division

□ L'opérateur division n'existe pas en SQL !



□ Par une **double négation d'existence**

- Tout simplement ;-)

La Division

- L'opérateur division n'existe pas en SQL !
 - "Quels sont les membres ayant opéré une ou plusieurs réservations sur tous les courts ?"

- Recours à une **double négation d'existence**
 - "Quels sont les membres tels qu'il n'existe pas de courts
 - ⇔
 - tels qu'il n'existe pas de réservation de **ces courts là** opérées par **ces membres là**"
 - qui n'aient pas été réservés par **ces membres là**"

La Division

- "Quels sont les membres tels qu'il n'existe pas de courts tels qu'il n'existe pas de réservation de ces courts là opérées par ces membres là"

```
Select NOM, PRENOM
From MEMBRE
Where Not Exists (
  Select NUMERO#
  From COURT
  Where Not Exists (
    Select NUMERO#
    From RESERVATION
    Where COURT## = COURT.NUMERO#
    And MEMBRE## = MEMBRE.NUMERO# ) )
```

113

Principes Des Bases De Données



SQL – LMD

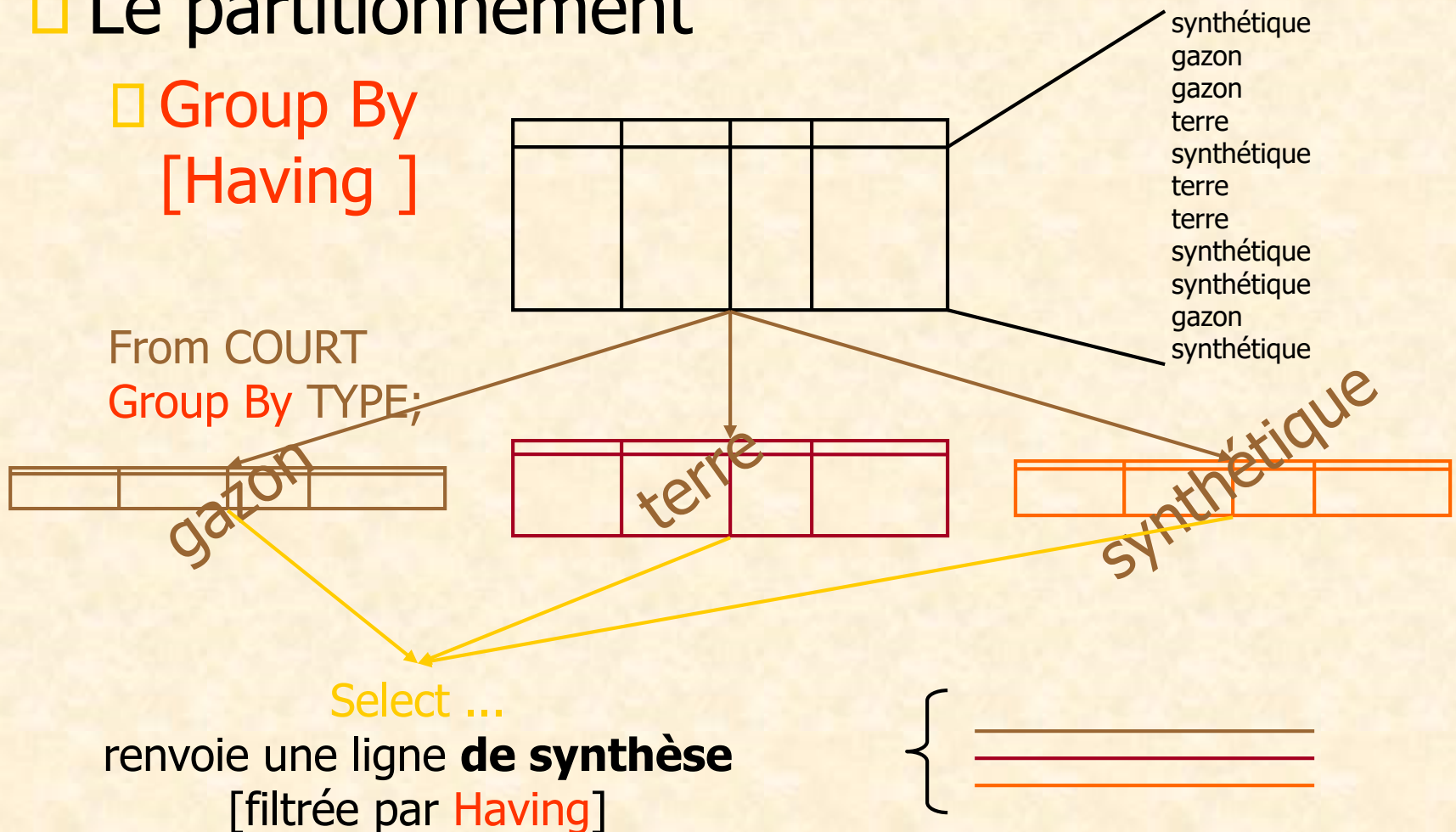
L'interrogation

**Les opérateurs
supplémentaires**

L'instruction Select

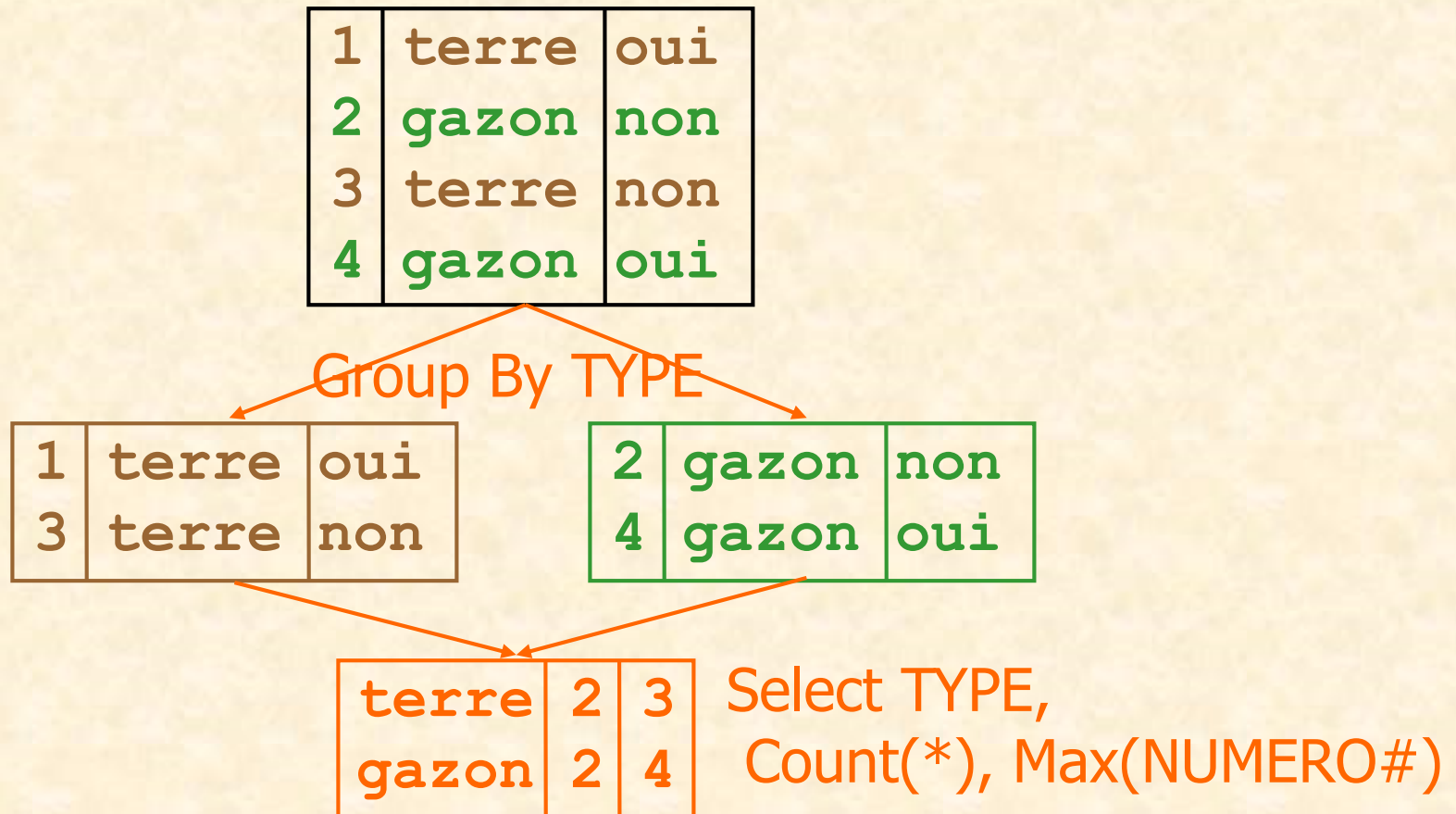
□ Le partitionnement

□ Group By [Having]



L'instruction Select

□ Le partitionnement



L'instruction Select

□ Le partitionnement

- Comme le Select ne renvoie qu'une seule ligne de synthèse "par sous-table"...
- L'on ne peut demander que les colonnes du partitionnement ou des calculs verticaux !

```
Select TYPE, Count(TYPE)  
From COURT  
Group By TYPE;
```

```
Select TYPE, Count(*)  
...
```

```
Select NUMERO#, TYPE, Count(TYPE)  
From COURT  
Group By TYPE;
```

```
Select NUMERO#, TYPE, Count(TYPE)  
*
```

ERREUR à la ligne 1 :

ORA-00979: N'est pas une expression GROUP BY

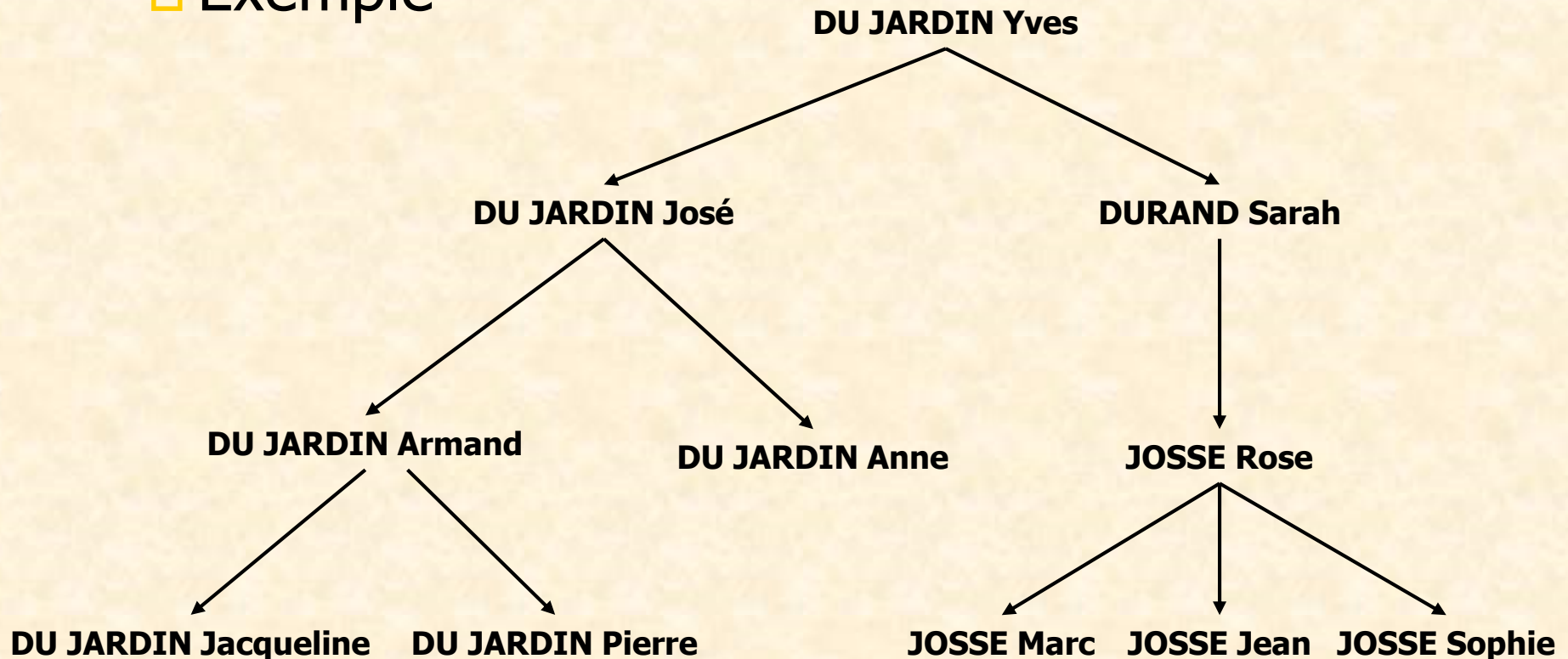
L'instruction Select

- Le traitement des structures d'arbre
 - [Where **Level** θ expr]
 - [CONNECT BY col = **PRIOR** col]
 - [START WITH <condition>]]
- **Prior** identifie le précédent dans la liste
- **Level** indique le niveau dans l'arbre
- Détection des boucles, interruption et message
- NE peut PAS s'opérer sur une jointure

L'instruction Select

□ Le traitement des structures d'arbre

□ Exemple

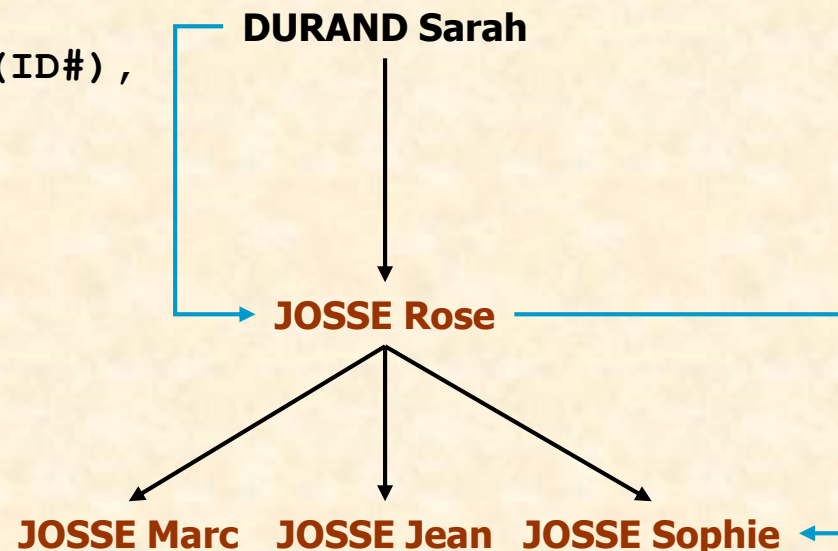


L'instruction Select

- Le traitement des structures d'arbre
 - Quels sont les descendants de DURAND Sarah ?

```
Create Table FAMILLE (  
  ID#          Number(2) Primary Key,  
  NOM          VarChar2(30),  
  PRENOM       VarChar2(20),  
  PARENT## Number(2) References FAMILLE(ID#),  
  Unique(NOM, PRENOM) );
```

```
Select LPAD(' ', LEVEL+1) ||  
      NOM || ' ' || PRENOM  
From FAMILLE  
Where Level > 1  
Connect By PARENT## = Prior ID#  
Start With  NOM = 'DURAND'  
           And PRENOM = 'Sarah';
```



L'instruction Select

- Le traitement des structures d'arbre
 - Quels sont les parents de JOSSE Marc ?

```
Select LPAD(' ',LEVEL+1)
      || NOM || ' ' || PRENOM "NOM PRENOM"
From FAMILLE
Where Level > 1
Connect By ID# = Prior PARENT##
Start With  NOM = 'JOSSE' And PRENOM = 'Marc';
```

Le père de

DU JARDIN Yves

DURAND Sarah

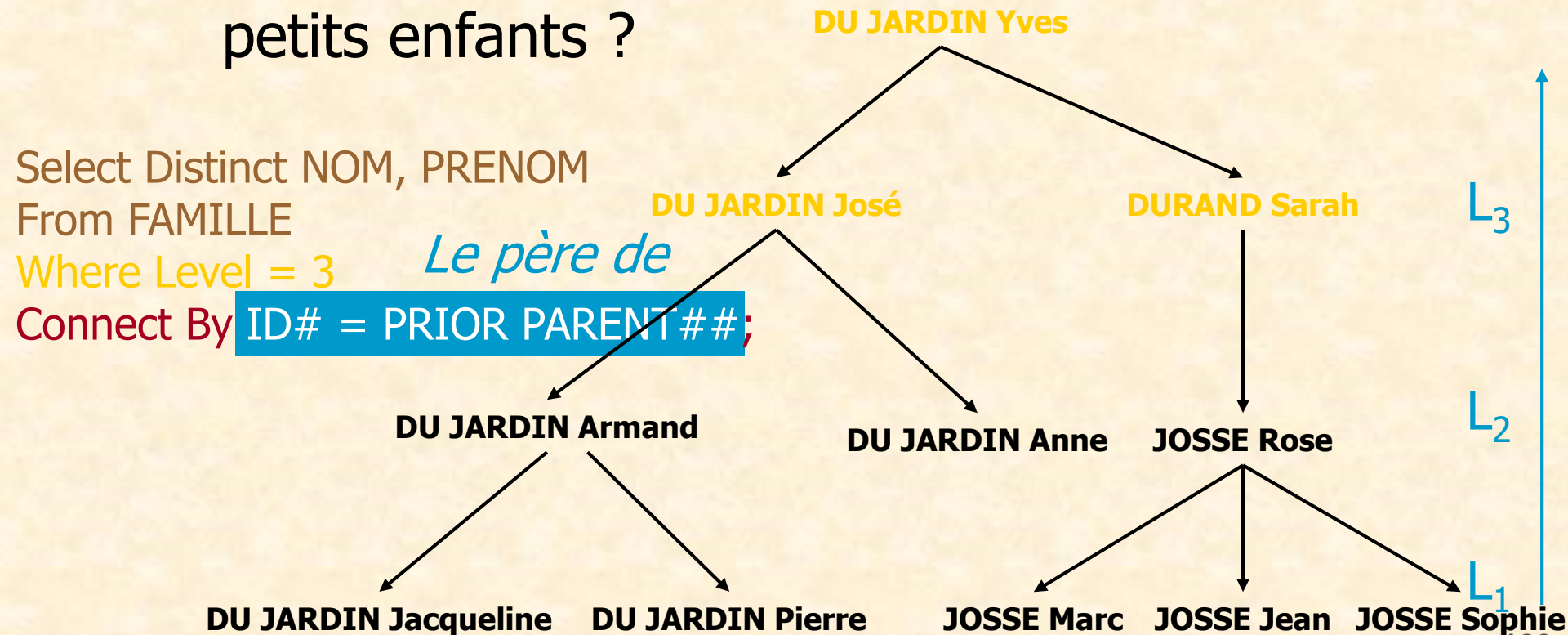
JOSSE Rose

JOSSE Marc



L'instruction Select

- Le traitement des structures d'arbre
 - Quels sont les membres de la famille ayant des petits enfants ?



Principes Des Bases De Données



SQL – LMD

L'interrogation

Les fonctions

L'instruction Select

- Les expressions et fonctions (cf. la doc !)
 - Concaténation : `||`
 - Fonctions numérique
 - Fonctions date
 - `+ N` la date + `N` jours
 - `Add_Months(la_date, N)`
 - `To_date('15/09/2003','dd/mm/yyyy')`
 - `Nvl(expr1,expr2)`
 - `Decode(expr,{val1,resultat1},valDéfaut)`
 - ...

Principes Des Bases De Données



SQL – LMD

Insertion

Mise à jour

Suppression

L'insertion

□ S'opère par l'instruction **Insert Into**

Insert Into table [(attribut₁, ...)]
Values (... , ...) | Select...

```
Insert Into MEMBRE Values (1, 'PEEL', 'EMMA', '04 93 95 12 48', 'Cannes',  
                           28, 'F', 10.7);
```

```
Insert Into MEMBRE (NUMERO#, NOM, PRENOM) Values (2, 'STEED', 'JOHN')
```

```
Insert Into RESERVATION
```

```
  Select 100, 1, MEMBRE.NUMERO#, TO_DATE('15/09/2003','dd/mm/yyyy'),  
         10, 12, 2
```

```
From MEMBRE
```

```
Where NOM='DUPONT' And PRENOM='CHARLES';
```

L'insertion



□ Les séquences Oracle

- Permettent la génération automatique de compteurs

 - Utile pour les clés primaires

- Create Sequence COURT_NUMERO# Start With 1 Increment By 1 MaxValue 20;

Insert Into COURT Values (COURT_NUMERO#.NextVal, 'terre', 'non');

La mise à jour



□ S'opère par l'instruction **Update**

Update table

Set attribut=valeur | expression

Where condition

Update COURT

Set DOUBLE='oui'

Where NUMERO# = 3;

La suppression

□ S'opère par l'instruction **Delete**

Delete From table
[Where condition]

Delete From RESERVATION;

Delete From COURT
Where TYPE='gazon';

□ Il existe aussi

□ **Truncate** supprime toutes les lignes

□ **Drop** supprime la table

Le dictionnaire de données Oracle



- Il est stocké dans la base de données
- Des vues sont définies par dessus pour en faciliter l'accès
 - e.g. {USER_, ALL_, DBA_}{TABLES, INDEXES, SEQUENCES, TRIGGERS}
- La commande **DESCRibe**

Principes Des Bases De Données



SQL - LDD

Un schéma de données relationnel



- Domaines
- Relations et attributs
- Règles de gestion

Les commandes du LDD



- Create Domain
 - Généralement nom implanté
- Create Table
 - Attribut : type, check, default, not null
 - Artefact syntaxique usuel du concept sémantique de domaine
 - Des Index sont associés implicitement/explicitement aux tables
- Create Trigger

SQL Oracle



- Plus largement, ORACLE permet la manipulation des éléments suivants
 - Tables, Vues, Index
 - Séquences
 - Synonymes
 - Clusters
 - Espaces logiques de stockage
 - Triggers

Les commandes du LDD

□ Via les commandes

- CREATE : création d'un élément
- ALTER : modification d'un élément
- TRUNCATE : supprimer les lignes d'une table
- DROP : supprimer un élément
- RENAME : renommer un élément
- Un élément est un/une
 - Table, vue, index, séquence, synonyme, cluster, espace logique de stockage, trigger
 - Rôle, utilisateur (LCD)

Création d'une table

- Les types d'attributs les plus utilisés
 - CHARACTER(n), CHAR(n)
 - CHAR(n)
 - CHARACTER VARYING(n), CHAR VARYING(n)
 - VARCHAR(n), Oracle recommande VARCHAR2(n)
 - NUMERIC(p,s), DECIMAL(p,s)
 - NUMBER(p,s)
 - p nombre total de chiffres (i.e. avant et après la virgule)
 - s nombre de chiffres après la virgule
 - Si non précisé = 0

Création d'une table

- Les types d'attributs les plus utilisés
 - INTEGER, INT, SMALLINT
 - NUMBER(38) !!!
 - FLOAT(b), DOUBLE PRECISION, REAL
 - NUMBER
 - A floating-point number with decimal precision 38 !!!
 - DATE

Création d'une table

- Les modèles de format
 - Chaînes de caractères utilisées avec TO_CHAR et TO_DATE pour spécifier le format de présentation d'une colonne DATE ou NUMBER
 - Ne modifie pas la représentation interne
- Exemples
 - SELECT TO_CHAR(comm, '\$9,990.99')
 - SET hiredate = TO_DATE('1998 05 20', 'YYYY MM DD')

Création d'une table



```
Create Table COURT (  
  NUMERO# Numeric(2),  
  TYPE VarChar2(13),  
  DOUBLE Character(3) );
```

?!

Prise en compte des règles de gestion



- Au travers de contraintes
 - Niveau colonne
 - Niveau Table
- Grâce à des triggers

Création d'une table



```
Create Table NOM_TABLE (  
    NOM_COLONNE typeDeDonnées  
        [DEFAULT expr]  
        [contrainteAuNiveauColonne],  
    ...  
    [contraintesAuNiveauTable]... );
```

```
Create Table NOM_TABLE As  
    sous-requête;
```

Création d'une table

```
Create Table COURT (  
    NUMERO# Numeric(2)  
    Constraint COURT_NUMERO_VALIDE  
        Check (1 <= NUMERO# And NUMERO# <= 20)  
    Constraint COURT_NUMERO_PK Primary Key,  
  
    TYPE VarChar2(13)  
    Constraint COURT_TYPE_VALIDE  
        Check ( TYPE In ('terre', 'gazon', 'synthétique') )  
    Constraint COURT_TYPE_NOT_NULL Not Null,  
    DOUBLE Character(3)  
    Constraint COURT_DOUBLE_VALIDE  
        Check ( DOUBLE In ('oui', 'non') )  
    Constraint COURT_DOUBLE_NOT_NULL Not Null  
);
```

Création d'une table

```
Create Table MEMBRE (  
    NUMERO# Numeric(3)  
    Constraint MEMBRE_NUMERO_VALIDE  
        Check (1 <= NUMERO# And NUMERO# <= 600)  
    Constraint MEMBRE_NUMERO_PK Primary Key,  
    NOM VarChar2(30)  
    Constraint MEMBRE_NOM_NOT_NULL Not Null  
    Constraint MEMBRE_NOM_MAJUSCULES Check ( NOM = Upper(NOM) ),  
    ...  
    TELEPHONE Character(14)  
    Constraint MEMBRE_TELEPHONE_VALIDE Check ( Length(TELEPHONE) = 14  
        And Substr(TELEPHONE, 1, 1) = '0' And (Substr(TELEPHONE, 2, 1) In  
        ('1', '2', '3', '4', '5')) And (Substr(TELEPHONE, 3, 1) = ' '  
        And (Substr(TELEPHONE, 6, 1) = ' ') And (Substr(TELEPHONE, 9, 1) = ' '  
        And (Substr(TELEPHONE, 12, 1) = ' ') ),  
    ...  
    -- Contrainte de Table  
    Constraint MEMBRE_UNIQUE Unique(NOM, PRENOM, TELEPHONE) );
```

Création d'une table

```
Create Table RESERVATION (  
  NUMERO# Numeric(4) ...,  
  COURT## Numeric(2)  
    Constraint RESERV_COURT#_FK  
      References COURT(NUMERO#) On Delete Cascade  
    Constraint RESERV_COURT##_NOT_NULL Not Null,  
  MEMBRE## Numeric(3)  
    Constraint RESERV_MEMBRE#_FK  
      References MEMBRE(NUMERO#) On Delete Cascade  
    Constraint RESERV_MEMBRE##_NOT_NULL Not Null,  
  LA_DATE Date Default SYSDATE  
    Constraint RESERV_LA_DATE_NOT_NULL Not Null,  
  ...  
  -- Contrainte de Table  
  Constraint HEURES_RESERV_VALIDES  
    Check ( HEURE_DEBUT < HEURE_FIN ) );
```

Création d'une vue



- Une vue est une couche d'abstraction
 - C'est une requête
 - Elle ne contient donc pas de données
- Intérêt
 - Faciliter l'accès aux données
 - Réguler l'accès aux données
 - E.g. les utilisateurs ont seulement des privilèges sur les vues
 - "Indépendance" entre traitements et données

Création d'une vue



Create View COURT_TERRE As

Select NUMERO#, DOUBLE

From COURT

Where TYPE='terre';

Create View RESERVATION_TERRE As

Select RESERVATION.NUMERO#, NOM, PRENOM, COURT##, LA_DATE,
HEURE_DEBUT, HEURE_FIN, NB_JOUEURS

FROM COURT, MEMBRE, RESERVATION

Where TYPE='terre' And COURT## = COURT.NUMERO#

And MEMBRE## = MEMBRE.NUMERO#;

Drop View COURT_TERRE;

Restriction sur les vues

- Pas d'ORDER BY
- Insertions, mises à jours et suppressions impossibles si
 - La vue contient une jointure, des opérateurs ensemblistes, des fonctions de groupe, les clauses Group By, Connect By ou Start With et l'opérateur Distinct !

Manipulation des vues



- Une vue se manipule comme une table avec les commandes SQL
 - Select
 - Insert
 - Delete
 - Update
- À nouveau, les données ne sont pas dans la vue

Enfin, il y a aussi



- Create [Unique] | Drop Index
- Alter Table
- Create [Public] Synonym
- Create Trigger
- ...

Principes Des Bases De Données



SQL - LCD

Les commandes du LCD



- Création d'utilisateurs
- Privilèges et Rôles

Les commandes du LCD



□ Création d'utilisateurs

- Create User NOM Identified By MOT_PASSE;

- + les tablespaces (cf. l'architecture d'Oracle)

□ Notion de schéma Oracle

Les commandes du LCD



□ Les privilèges

□ Système

- E.g. Create Table, Create Session, Create Index

□ Objet

- Select On COURT

- Update (NOM) On MEMBRE

□ Les rôles

- Regroupement de privilèges

- Create Role

Les commandes du LCD



- Affectation de privilèges ou rôles aux utilisateurs ou rôles
 - Grant QUOI To QUI [With Grant Option]

- Retrait de privilèges ou rôles
 - Revoke QUOI From QUI

Principes Des Bases De Données



Les transactions

Une transaction

□ Une capsule de transition d'état du SI respectant les propriétés **ACID**

□ **A**tomacité

□ **C**ohérence

□ **I**solation

□ **D**urabilité

□ Une transaction a

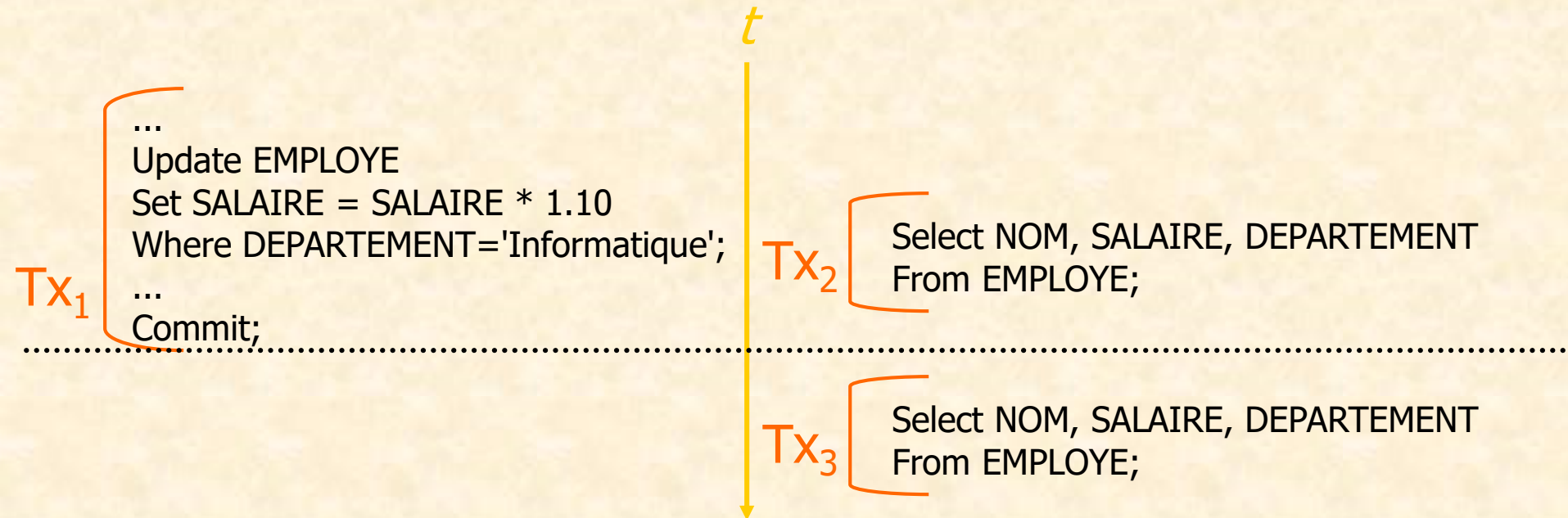
□ Un début

□ Une fin : validation ou annulation



Transactions concurrentes

□ Soit les transactions suivantes

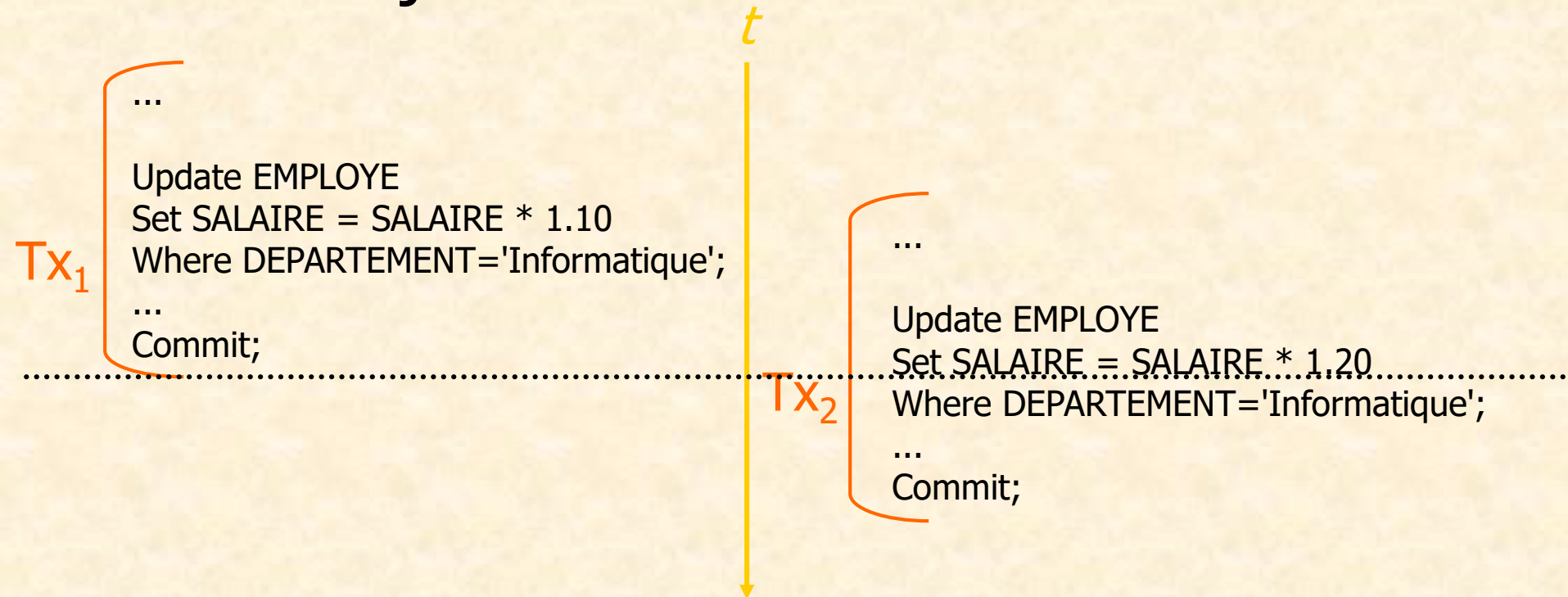


□ Salaire des informaticiens en Tx_2 et Tx_3 ?

□ Propriété d'**I**solation !

Transactions concurrentes

□ Mises à jour concurrentes

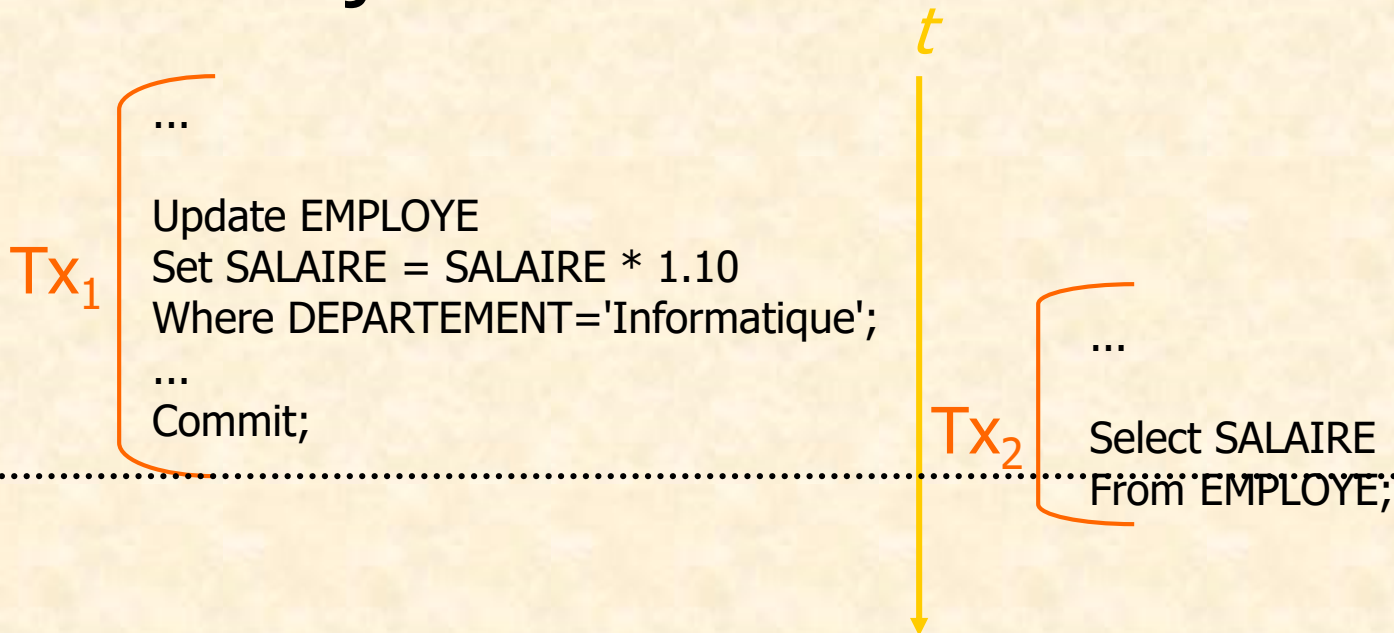


□ Danger de perte de mises à jour

□ Salaire + 20% ou + 30% ?

Transactions concurrentes

□ Mise à jour et lecture concurrentes

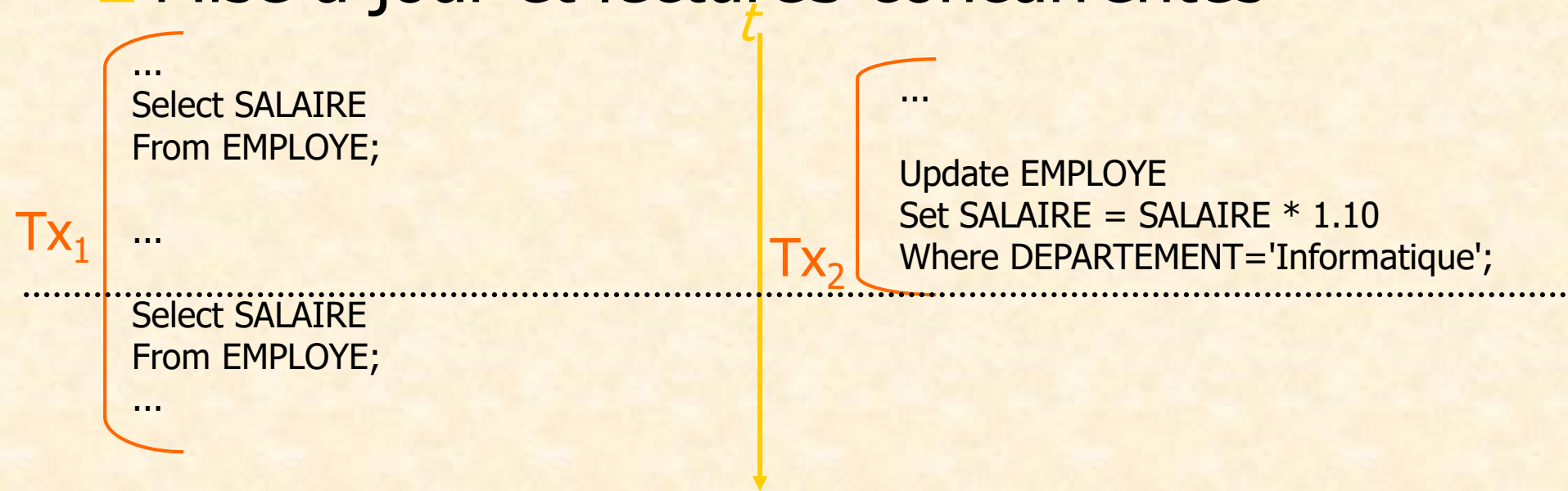


□ En Tx_2 que vaut le salaire : +0%, +10% ?

□ Et plus généralement, l'impaticence produirait-elle une information "obsolète" ?

Transactions concurrentes

□ Mise à jour et lectures concurrentes



□ Lecture non reproductible, si Tx_2 termine avant le deuxième Select²

Transactions concurrentes

□ Mises à jour concurrentes sur ensemble disjoints au départ t

...

Update EMPLOYE
Set DEPARTEMENT='Informatique'
Where MATRICULE# In (...);

...

Commit;

...


Update EMPLOYE
Set SALAIRE = SALAIRE * 1.05
Where DEPARTEMENT='Informatique';

...

Commit;

□ Les mutés seront-ils augmentés ?

Transactions concurrentes



- Les transactions concurrentes peuvent poser des problèmes
 - Mises à jour perdues
 - Lectures "non logiques"
- La précaution retenue par les SGBD est le verrouillage
 - Assure la **sérialisation** des opérations concurrentes sur les données
 - Et donc la propriété de **C**ohérence

Transactions concurrentes, les verrous



- Variable d'état pour l'accès aux données
 - Au niveau
 - Table
 - Ligne
 - En mode
 - Partagé (i.e. share)
 - Une Tx concurrente peut lire, mais pas mettre à jour
 - Exclusif (i.e. exclusive)
 - Impossible de lire ou mettre à jour depuis une autre Tx

Transactions concurrentes, les verrous

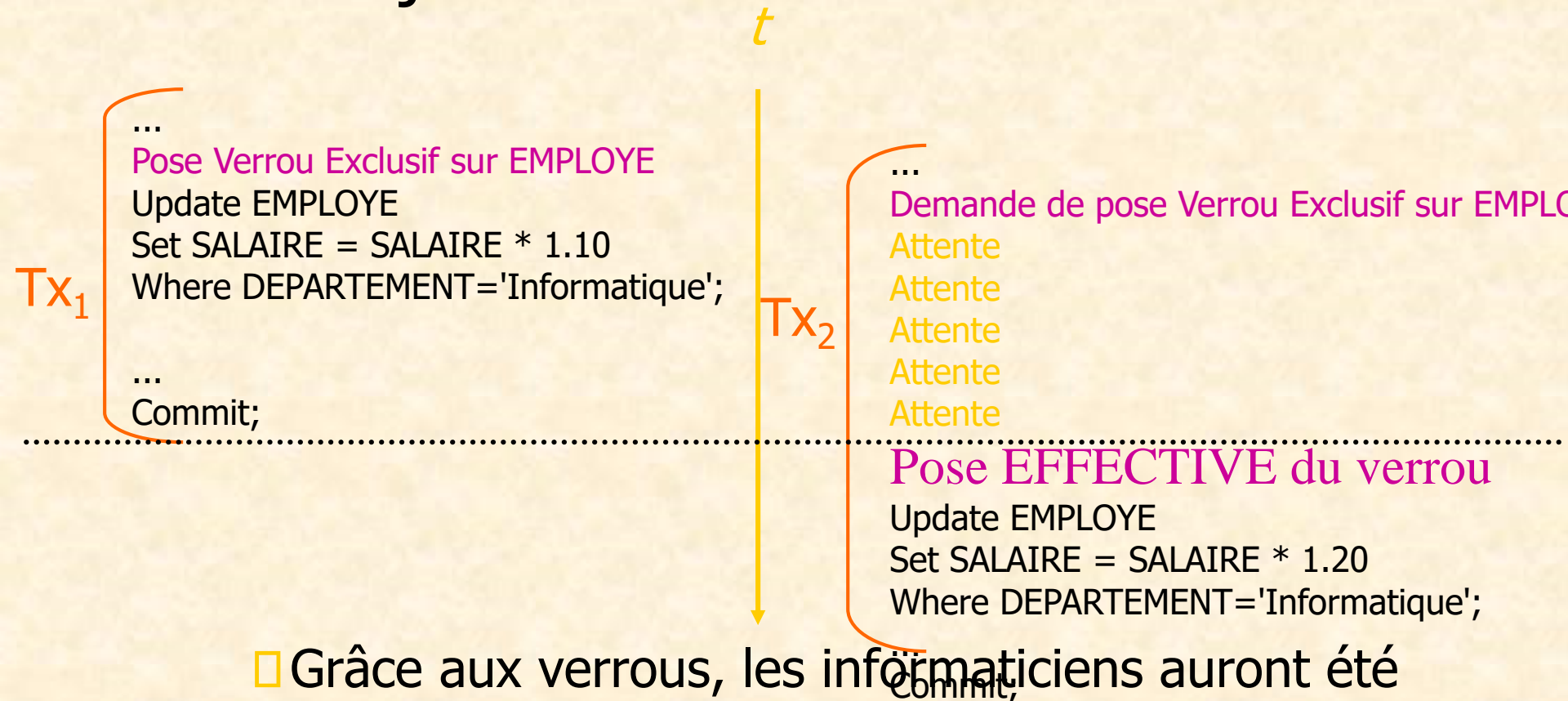


□ Principe

- Une Tx pose un verrou **partagé** préalablement à la **lecture** d'une donnée
- Une Tx pose un verrou **exclusif** préalablement à la **mise à jour** d'une donnée

Transactions concurrentes, les verrous

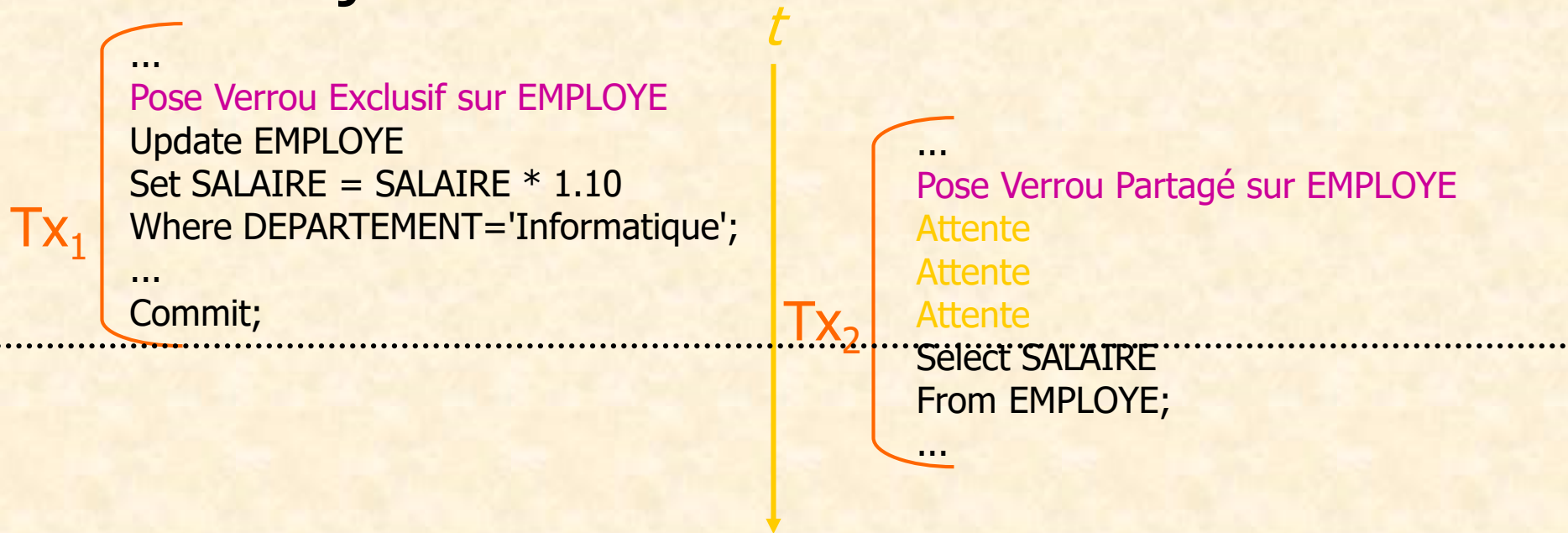
□ Mises à jour concurrentes



□ Grâce aux verrous, les informaticiens auront été augmentés de 10% + 20%

Transactions concurrentes

□ Mise à jour et lecture concurrentes



□ Grâce aux verrous, je vois l'augmentation des informaticiens

Transactions concurrentes

□ Mises à jour concurrentes sur ensemble disjoints au départ t

...
Pose Verrou Exclusif sur EMPLOYE
Update EMPLOYE
Set DEPARTEMENT='Informatique'
Where MATRICULE# In (...);
...
Commit;

...
Pose Verrou Exclusif sur EMPLOYE
Attente
Attente
Attente
Update EMPLOYE
Set SALAIRE = SALAIRE * 1.05
Where DEPARTEMENT='Informatique';
...
Commit;

□ Grâce aux verrous, les mutés sont augmentés

Une transaction sous Oracle

- Début implicite
 - I.e. fin de la transaction précédente !

- Fin
 - Explicite
 - `Commit`
 - `Rollback`
 - Implicite (validation)
 - Une commande du LDD, déconnexion

Une transaction Oracle pose des verrous



- Verrouillage au niveau

 - Table

 - Ligne

- Verrouillage

 - Implicite

 - Explicite

Principes des bases de données



**Introduction à
l'architecture du SGBD
Oracle**

Introduction à l'architecture du SGBD Oracle

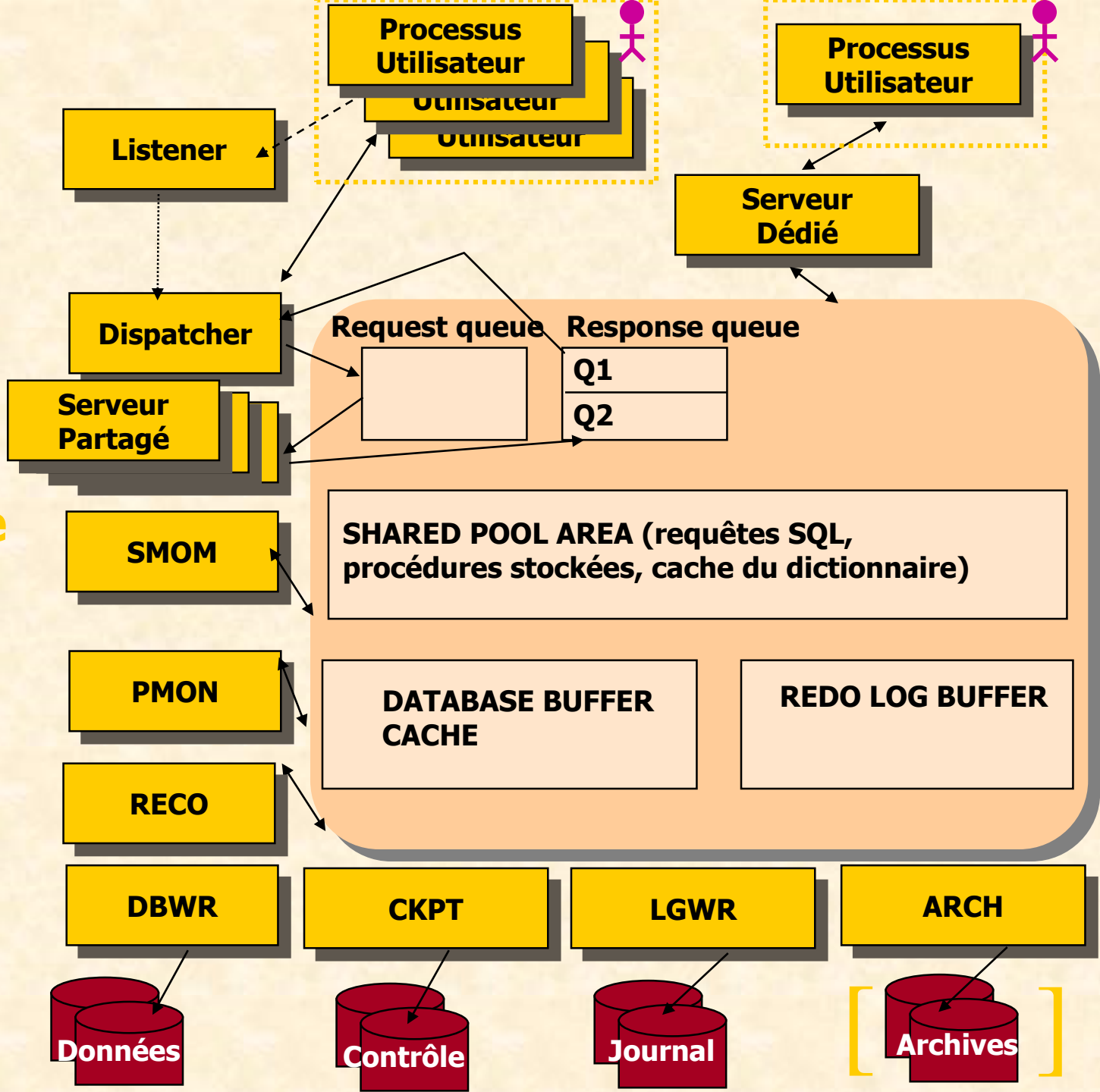


- Oracle Technology Network
 - <http://otn.oracle.com>

- Une instance Oracle
 - Gère une base de données Oracle
 - L'instance doit être démarrée afin de pouvoir utiliser la base de données

L'instance

La BD



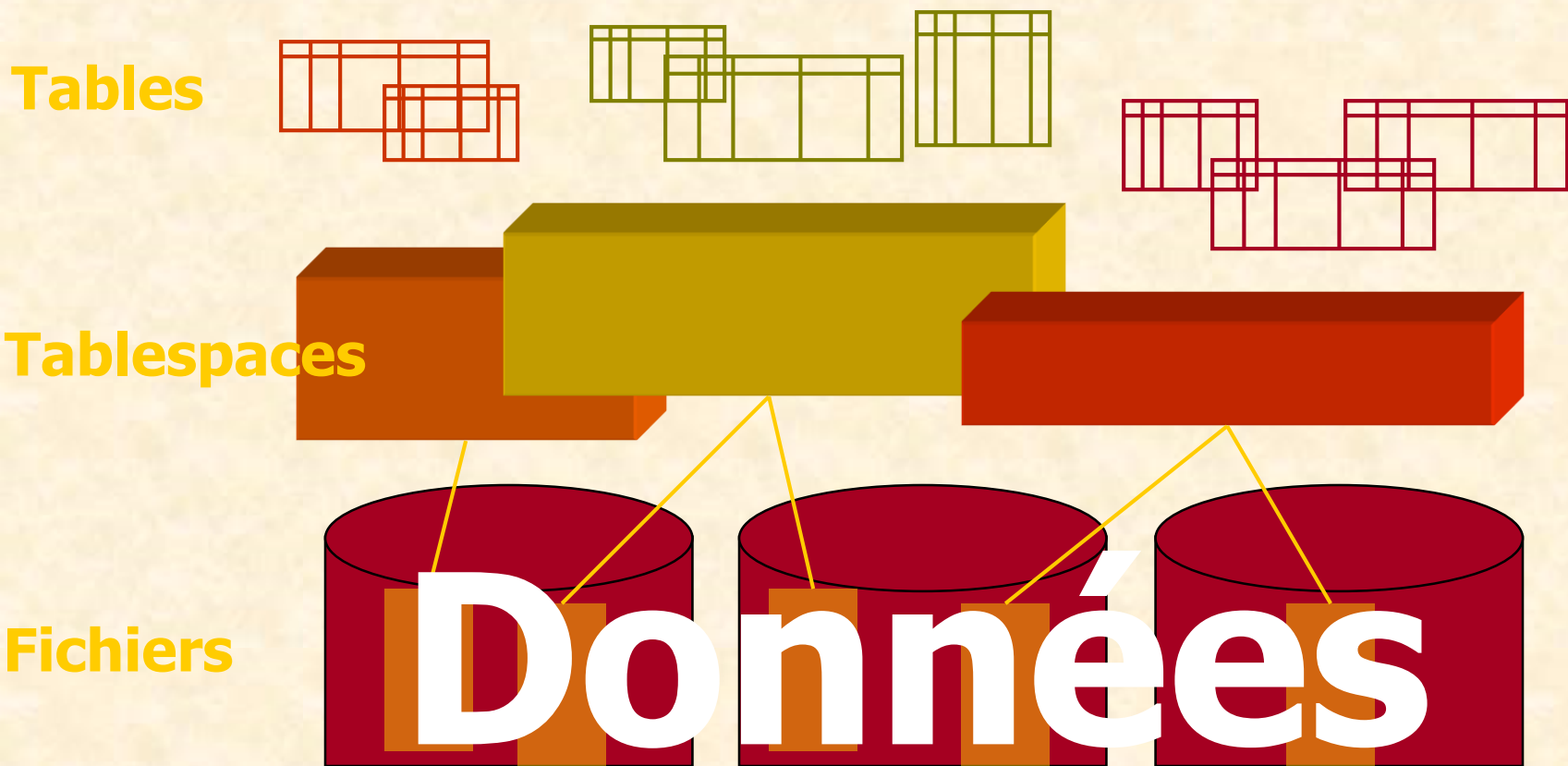
Introduction à l'architecture du SGBD Oracle




- Création d'une base de données
 - `Create Database...`
 - Deux utilisateurs sont créés
 - `SYS`
 - `SYSTEM`
- La base de données contiendra *N* schémas
 - Autant qu'il y aura d'utilisateurs
 - `Create User...`

Introduction à l'architecture du SGBD Oracle

- Les Tablespaces : lien entre les niveaux physique et logique



Introduction à l'architecture du SGBD Oracle



- Les Tablespaces : lien entre les niveaux physique et logique

```
Create Tablespace TS_USERS Datafile  
    'E:\BdOracle\816\DatabaseFiles\users01.dbf' size 25M;
```

```
Create Table FOO (  
    X Date )  
Tablespace TS_USERS;
```

Introduction à l'architecture du SGBD Oracle

□ Création d'utilisateur

```
Create User TOTO Identified By TITI  
Default TableSpace TS_USERS  
Quota Unlimited On TS_USERS  
Temporary TableSpace TS_TEMP  
Quota Unlimited On TS_TEMP;
```

□ Affectation de "permissions"

```
Grant Connect, Resource To TOTO;  
Revoke Unlimited TableSpace From TOTO;  
Grant SELECT_CATALOG_ROLE To TOTO;
```

Principes des Bases de Données

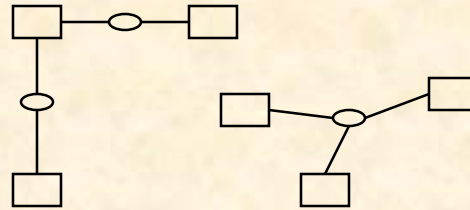


La conception d'un schéma relationnel :

- Les liens**
- Comment obtenir le schéma ?**

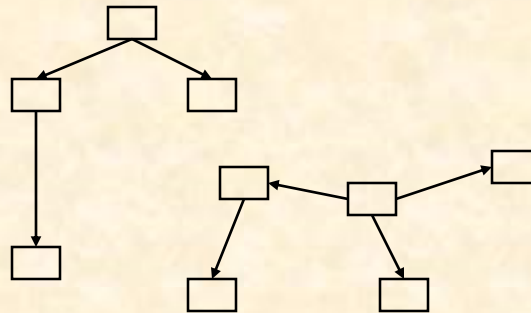
Concept° d'un Sch. Données : 3 niveaux (de formalisme)

Conceptuel



e.g. Entité-Association

Logique



e.g. Relationnel

Physique

Create Table ...

e.g. SQL

Principes des bases de données



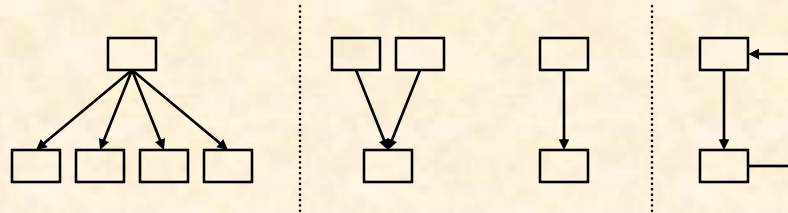
Les liens **identifiés par le
modèle relationnel...**

et le lien **implanté...**

Liens identifiés...

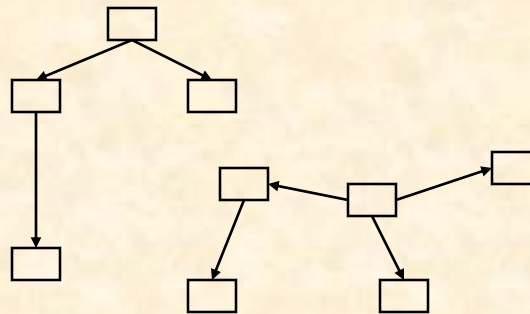
et lien implanté

Conceptuel
(liens identifiés)



1-N N-1 N-M

Logique
(lien implanté)



N-1

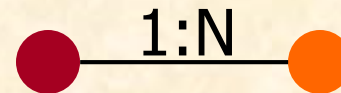
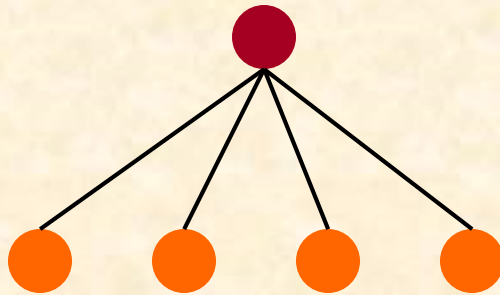
Physique

Create Table ...

N-1

Les liens

- Le type de liens 1:N (de 1 vers N)
- C'est le lien **hiérarchique** (descriptif)
 - I.e. 1 "chose" en décrit plusieurs : "Une mère a **plusieurs** enfants"



- "Un court de tennis est réservé par plusieurs membres"
 - I.e. Un court de tennis est utilisé par **plusieurs** réservations

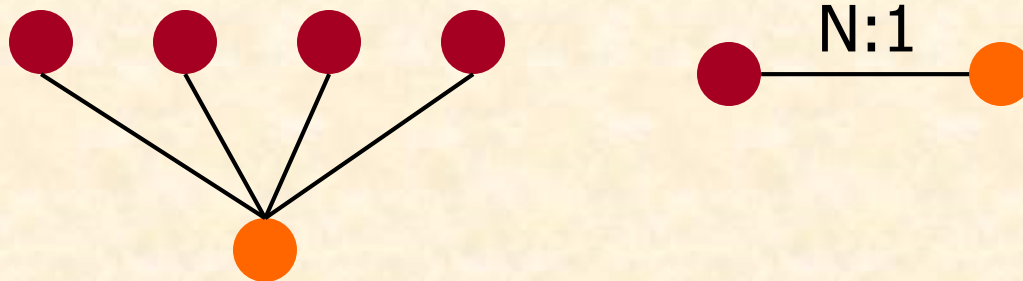
Les liens

□ Le type de liens N:1 (de N vers 1)

□ C'est le lien **déterminant**

□ I.e. 1 ou N "choses" en déterminent une et une seule

- Lien hiérarchique inverse : "Un enfant a **une** seule mère "

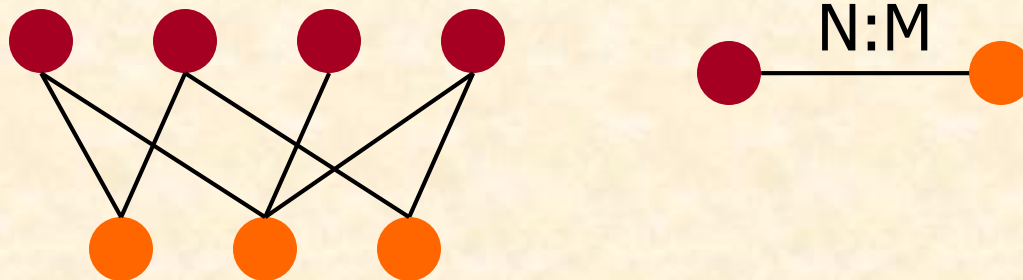


□ "Une réservation concerne un court de tennis"

- I.e. Une réservation concerne **un seul** court de tennis

Les liens

- Le type de liens N:M (de N vers M)
 - C'est le lien associatif
 - I.e. N "choses" sont associées à M : "Une facture spécifie **plusieurs** produits et un produit apparaît dans **plusieurs** factures"



- "Un membre du club de tennis réserve **plusieurs** courts et un court est réservé par **plusieurs** membres"

Les liens



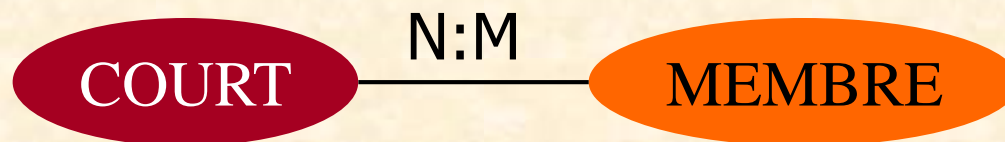
- Le type de liens 1:1 (de 1 vers 1)
 - Cas particulier (N peut valoir 1)

Le lien N:1

- Le lien N:1 (lien déterminant ou dépendance fonctionnelle) est le lien implanté par le modèle relationnel
 - Concept de la clé étrangère
 - $\text{RESERVATION}(\text{COURT}\#\#) \xrightarrow{\text{N:1}} \text{COURT}(\text{NUMERO}\#)$
 - Le seul outil pour représenter les liens identifiés
 - N:1 : OK !
 - 1:N ?
 - L'inverse du N:1, OK !

Le lien N:1

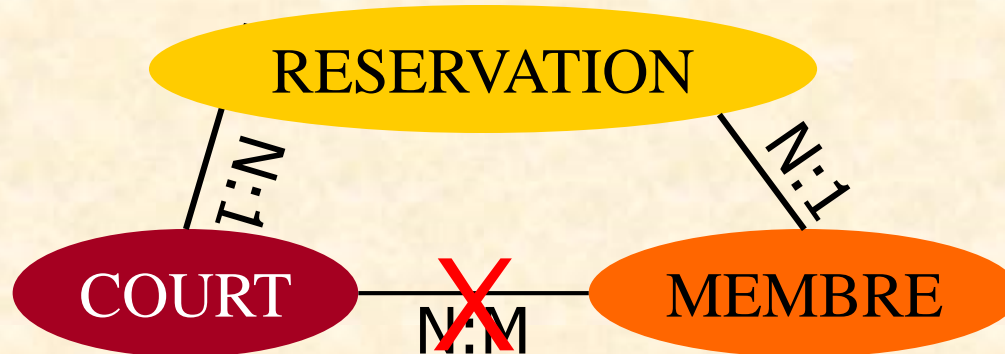
- Comment représenter le lien N:M avec le lien N:1 ?
 - "Un membre du club de tennis réserve **plusieurs** courts et un court est réservé par **plusieurs** membres"



- Ici : cas simple et manipulé jusqu'à présent !

Le lien N:1

- Donc solution évidente : 2 liens N:1
 - "Une réservation est faite par **un** membre"
 - RESERVATION(COURT##) → COURT(NUMERO#)
 - "Une réservation concerne **un** court de tennis"
 - RESERVATION(MEMBRE##) → MEMBRE(NUMERO#)

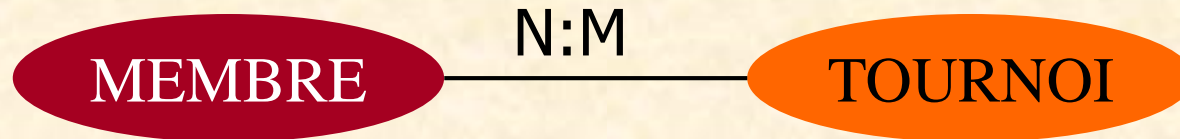


Passage de :
2 entités
à
3 entités

Le lien N:1

- D'une manière générale
 - Identifier OU créer une nouvelle entité pour casser le lien N:M en des liens N:1 (CÉ→CP)

?!



?!



Principes Des Bases De Données



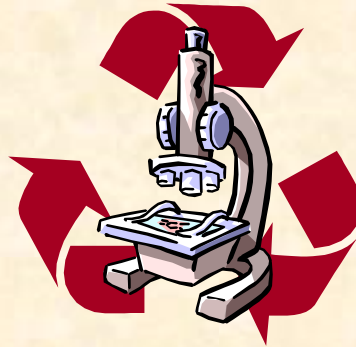
Comment obtenir un schéma relationnel ?

Comment obtenir un schéma relationnel ?

- Analyser l'univers réel, pour obtenir un ensemble de relation reliées entres-elles par des liens N:1



Univers réel



Modélisation
Relationnelle

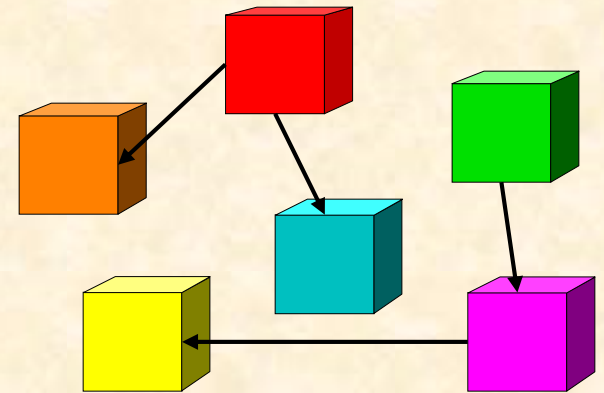
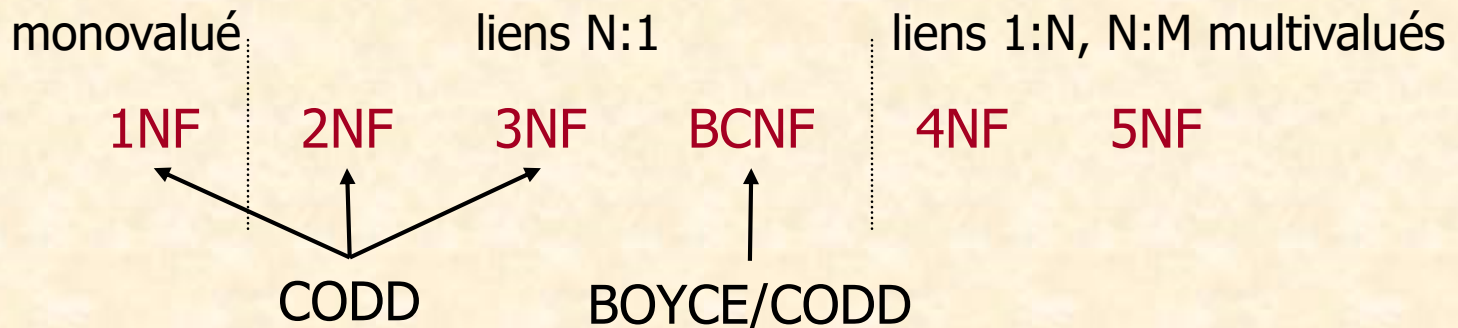


Schéma
Relationnel

Comment obtenir un schéma relationnel ?

- Schéma relationnel = N relations, **évitant** les **redondances**, les **problèmes de mise à jour et de suppression**
- But de la normalisation des relations par l'application des formes normales



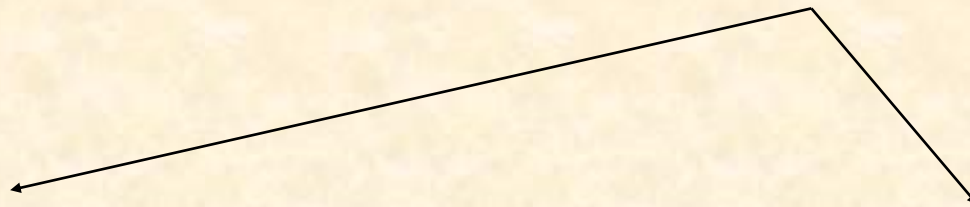
Les Formes Normales

□ 1 NF

- Les attributs de la relation contiennent des valeurs atomiques, ils sont **monovalués**
 - I.e. aucun domaine sur lequel est défini un attribut n'est lui même une relation

~~1 NF~~

EMPLOYEE(EMPID, NOM, PRENOMS, ENFANTS, DATENAISS)



eEMPLOYEE(empid, Nom, datenaiss)

Enfant(enfantid, EMPid, nom, dateNaiss)
PRENOMS(Empid, prenom)

Les Formes Normales

□ 2 NF : 1 NF +

- Aucun attribut non clé dépend seulement d'une partie de la CP
- Implicite si CP mono-formée ou multi-formée par tous les attributs

~~2 NF~~

PRODUIT(NOM, FOURNISSEUR, PRIX, ADRESSE)

(ARTICLE, FOURNISSEUR) → PRIX
FOURNISSEUR → ADRESSE

FOURNISSEUR (NOM, ADRESSE)

PRODUIT(NOM, FOURNISSEUR, PRIX)

Les Formes Normales

□ 3 NF : 2 NF +

- Aucun attribut non clé ne dépend d'un attribut non clé (**pas de relation dans une relation**)
 - Élimination des redondances dues aux transitivités

~~3 NF~~

PRODUIT(RÉF, LIBELLÉ, QT_STOCK, FOURNISSEUR, TÉLÉPHONE)

FOURNISSEUR → TÉLÉPHONE

PRODUIT(RÉF, LIBELLÉ, FOURNISSEUR, QT_STOCK)

FOURNISSEUR(NOM, TÉLÉPHONE)

Les Formes Normales

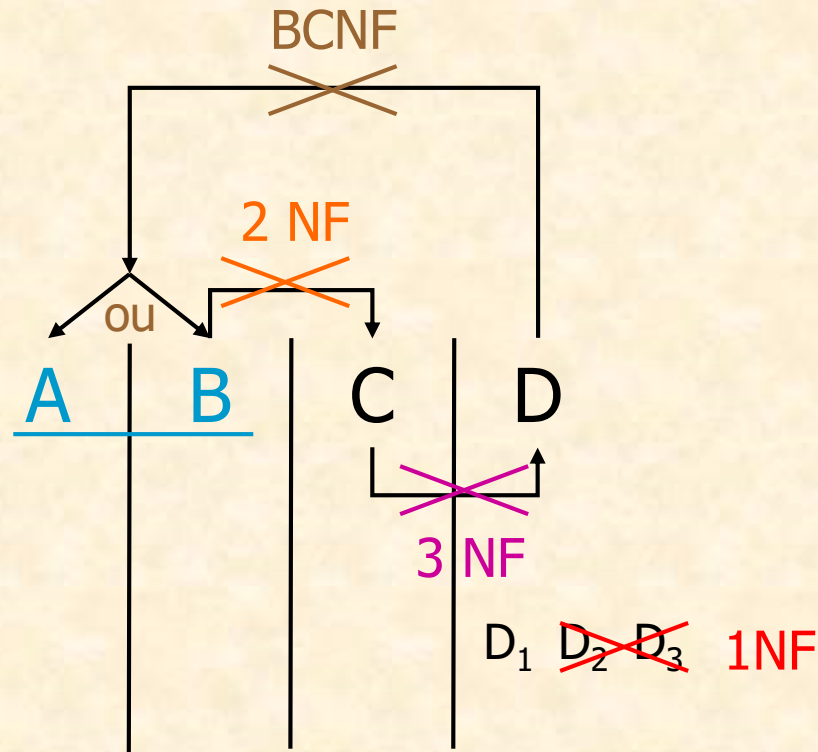


□ BCNF : 3 NF +

- Aucun attribut non clé ne détermine un attribut
CP : **seule les CP déterminent des attributs**
- Les cas de tables modélisées et transformées en 3FN non déjà en BCNF sont très rares
- Un modèle relationnel en BCNF est considéré comme étant de qualité suffisante pour une implantation !

1NF, 2NF, 3NF et BCNF

Résumé



Les dépendances multivaluées et la 4ème forme normale

Soit ETUDIANT = (Ne, Cours, Sport)

Ne	Cours	Sport
100	Bases de données	Tennis
100	Bases de données	Football
100	Réseaux	Tennis
100	Réseaux	Football
200	Réseaux	Vélo

Il existe des redondances mais aucune df.

Ce schéma est indécomposable !



La notion de df est insuffisante et conduit à introduire la notion de dépendance multivaluée

Notion de dépendance multivaluée (DM)

- Soit $R(A_1, A_2, \dots, A_n)$ un schéma de relation, X et Y des sous-ensembles de A_1, A_2, \dots, A_n . On dit que $X \twoheadrightarrow Y$ (X multidétermine Y ou il existe une dépendance multivaluée (DM) de Y sur X) si, étant donné des valeurs de X , il y a un ensemble de valeurs de Y associées et cet ensemble est indépendant des autres attributs $Z = R - X - Y$ de R
- Une telle dépendance caractérise une indépendance entre 2 ensembles d'attributs (Y et Z) corrélés par un même 3ème X .

Les Formes Normales

- La 4NF : BCNF + *liens 1:N ou N:M*
 - Dépendances multivaluées INdépendantes
- Un schéma de relation est en 4ème FN ssi les seules DM élémentaires sont celles dans lesquelles une super-clé détermine un attribut.
- Une super-clé est un ensemble d 'attributs contenant une clé.
- Ainsi R n 'est pas en 4 FN si l 'on peut trouver une DM $X \twoheadrightarrow Y$ où X n 'inclut pas une clé de R. (ex : ETUDIANT n 'est pas en 4 FN ; Clé = (Ne, Cours, Sport) ; $Ne \twoheadrightarrow Cours$, $Ne \twoheadrightarrow Sport$)

Les Formes Normales

- La 5NF *liens 1:N ou N:M*
- Isolement des liens multivaluées DÉpendants
 - Cas où l'information peut être reconstruite à partir de "morceaux" d'informations plus petits présentant eux moins de redondance
- Supposons que
 - Des VRP représentent des sociétés différentes
 - Les sociétés fabriquent des produits génériques
 - Les VRP vendent des produits génériques

Les Formes Normales

□ La 5NF (suite)

□ Supposons que (suite)

- Si un VRP vend un produit générique γ d'une société σ , alors il vend aussi γ pour les autres sociétés $\lambda \neq \sigma$
 - Les sociétés représentées par un VRP et les produits vendus par ce VRP

R	VRP	SOCIETE	PRODUIT VENDU
	KRIZMANIC	Euro Labo	vaccin grippe
	KRIZMANIC	Euro Labo	vaccin tétanos
	KRIZMANIC	Pharma Plus	vaccin tétanos
	KRIZMANIC	Pharma Plus	vaccin grippe
	RAMARI	Euro Labo	vaccin grippe
	RAMARI	Pharma Plus	vaccin grippe

Les Formes Normales

□ La 5NF (suite)

□ Pour éviter les redondances, R se décompose par trois projections

R1

VRP	SOCIETE
KRIZMANIC	Euro Labo
KRIZMANIC	Pharma Plus
RAMARI	Euro Labo

R2

SOCIETE	PRODUIT
Euro Labo	vaccin grippe
Euro Labo	vaccin tétanos
Pharma Plus	vaccin tétanos
Pharma Plus	vaccin grippe

R3

VRP	PRODUIT_VENDU
KRIZMANIC	vaccin grippe
KRIZMANIC	vaccin tétanos
RAMARI	vaccin grippe

$R = [Jointure\ R1(SOCIETE=SOCIETE)R2] Jointure\ R3(PRODUIT=PRODUIT_VENDU)$

Les Formes Normales

□ La 5NF : 4 NF +

□ *"Les tuples de la relation ne peuvent pas être reconstruits à partir de projections (donc "morceaux" d'informations plus petits) évitant les redondances"*

□ Sauf bien sûr si les projections ont la même CP

- La décomposition n'ayant alors aucun intérêt !

□ R1, R2 et R3 sont en 5NF ; R NE l'est PAS

□ La 5NF ne se différencie de la 4NF que s'il existe des liens **multivaluées DÉpendants**

□ Sinon une relation en 4NF est aussi en 5NF

Les Formes Normales



- A Simple Guide to Five Normal Forms in Relational Database Theory
 - <http://home.earthlink.net/~billkent/Doc/simple5.htm>
 - Article publié/présenté dans
 - Communications of the ACM 02/83
 - IBM Technical Report TR03.159, 08/81
 - SHARE 03/84
 - "Parallel Architectures for Database Systems" - IEEE Computer Society Press 1989

Principes Des Bases De Données



**Exemple de processus
d'obtention d'un schéma
relationnel**

Exemple de processus d'obtention d'un schéma rel.



- L'obtention d'un schéma relationnel en 3NF ou BCNF peut se faire via la méthode de la construction de la couverture minimale en 5 étapes
 - Étape 1
 - Recensement des attributs potentiels en éliminant synonymes et polysèmes, via l'étude de documents, interviewes, etc.

Exemple de processus d'obtention d'un schéma rel.

□ Construction de la couverture minimale

□ Étape 2

□ Recensement des **Dépendances Fonctionnelles**, composées d'un seul attribut en partie gauche

ou $\left[\begin{array}{l} \bullet \text{ Représentation graphique} \\ \bullet \text{ Matrice des DF} \end{array} \right.$

□ Obtention de relations à attributs déterminants mono-formé

Exemple de processus d'obtention d'un schéma rel.

□ Construction de la couverture minimale

□ Étape 2 (suite)

□ La **matrice des DF** et son traitement

- La matrice se compose des attributs identifiés à l'étape 1
- Si attribut_a détermine attribut_b , la cellule contient un 1
 - Donc, pour tout $i=j$, $(\text{ligne}_i, \text{colonne}_j)=1$
 - Si deux colonnes identiques, en supprimer une au niveau des DF (ne pas supprimer l'attribut !)
- Les colonnes contenant plus d'un 1 sont des attributs déterminants
 - Créer la **matrice des attributs déterminants**, pour traiter leur(s) transitivité(s) éventuelle(s) ($N:1, N=1$)

Exemple de processus d'obtention d'un schéma rel.

□ Construction de la couverture minimale

□ Étape 2 (suite)

□ La **matrice des DF** et son traitement (suite)

- Traduire les DF identifiées par la matrice sur un graphe (i.e. rattacher les attributs déterminés aux attributs déterminants)
 - Pour supprimer les transitivités, procéder dans l'ordre inverse des poids des attributs déterminants mis en évidence par la matrice des attributs déterminants

Exemple de processus d'obtention d'un schéma rel.

□ Construction de la couverture minimale

□ Étape 3

- Identifier les attributs non encore affectés

□ Étape 4

- Pour les attributs identifiés à l'étape 3, rechercher les DF composées en partie gauche d'attribut **S** ($N:1, N>1$) déterminants identifiés à l'étape 2

□ Étape 5

- Pour les attributs non encore affectés, rechercher les DF où l'un de ces attributs apparaît en partie gauche ($N:1, N>1$)

Exemple de processus d'obtention d'un schéma rel.

□ Exemple d'application

- "C'est l'histoire de VRP qui visitent des clients afin que ceux-ci opèrent des commandes de produits. Les VRP ne se partagent pas les clients. À des fins de statistiques, la société doit connaître pour chaque jour ouvré la quantité commandée d'un produit. Pour le calcul des primes, il faudra connaître le CA des VRP. Les nom, noms et prénoms peuvent faire l'objet d'homonymes."

Exemple de processus d'obtention d'un schéma rel.

□ Exemple d'application

□ Étape 1

ADRESSE_CLIENT
ADRESSE_LIVRAISON
CA_VRP
DATE_COMMANDE
JOUR_OUVRE
LIBELLE_PRODUIT
MOBILE_VRP
NO_CLIENT
NO_COMMANDE
NO_PRODUIT
NO_VRP
NOM_DESTINATAIRE

NOM_VRP
PRENOM_VRP
PRIX_UNITAIRE
QT_COMMANDEE
QT_PRODUIT_COMMANDE_JOUR
QT_STOCK
RAISON_SOCIALE

Exemple de processus d'obtention d'un schéma rel.

□ Exemple d'application

□ Étape 2 : Matrice des DF

		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
ADRESSE_CLIENT	1	1							1	1										1
ADRESSE_LIVRAISON	2		1							1										
CA_VRP	3			1				1	1			1								1
DATE_COMMANDE	4				1					1										
JOUR_OUVRE	5					1														
LIBELLÉ_PRODUIT	6						1				1									
MOBILE_VRP	7							1	1	1		1								1
NO_CLIENT	8								1	1										1
NO_COMMANDE	9									1										
NO_PRODUIT	10										1									
NO_VRP	11							1	1	1		1								1
NOM_DESTINATAIRE	12									1			1							
NOM_VRP	13							1	1	1		1		1						1
PRENOM_VRP	14							1	1	1		1			1					1
PRIX_UNITAIRE	15										1					1				
QT_COMMANDÉE	16																1			
QT_PRODUIT_COMMANDÉ_JOUR	17																	1		
QT_STOCK	18										1									1
RAISON_SOCIALE	19								1	1										1
	Σ							5	8	11	4	5								8

Exemple de processus d'obtention d'un schéma rel.

□ Exemple d'application

□ Étape 2 (suite) : Matrice des attributs déterminants

NO_CLIENT
NO_COMMANDE
NO_PRODUIT
NO_VRP

	8	9	10	11
8	1	1		
9		1		
10			1	
11	1	1		1
Σ	2	3		

□ Pour obtenir le graphe des attributs déterminants, procéder dans l'ordre inverse des Σ (N:1, N=1)

NO_COMMANDE



NO_CLIENT



NO_PRODUIT

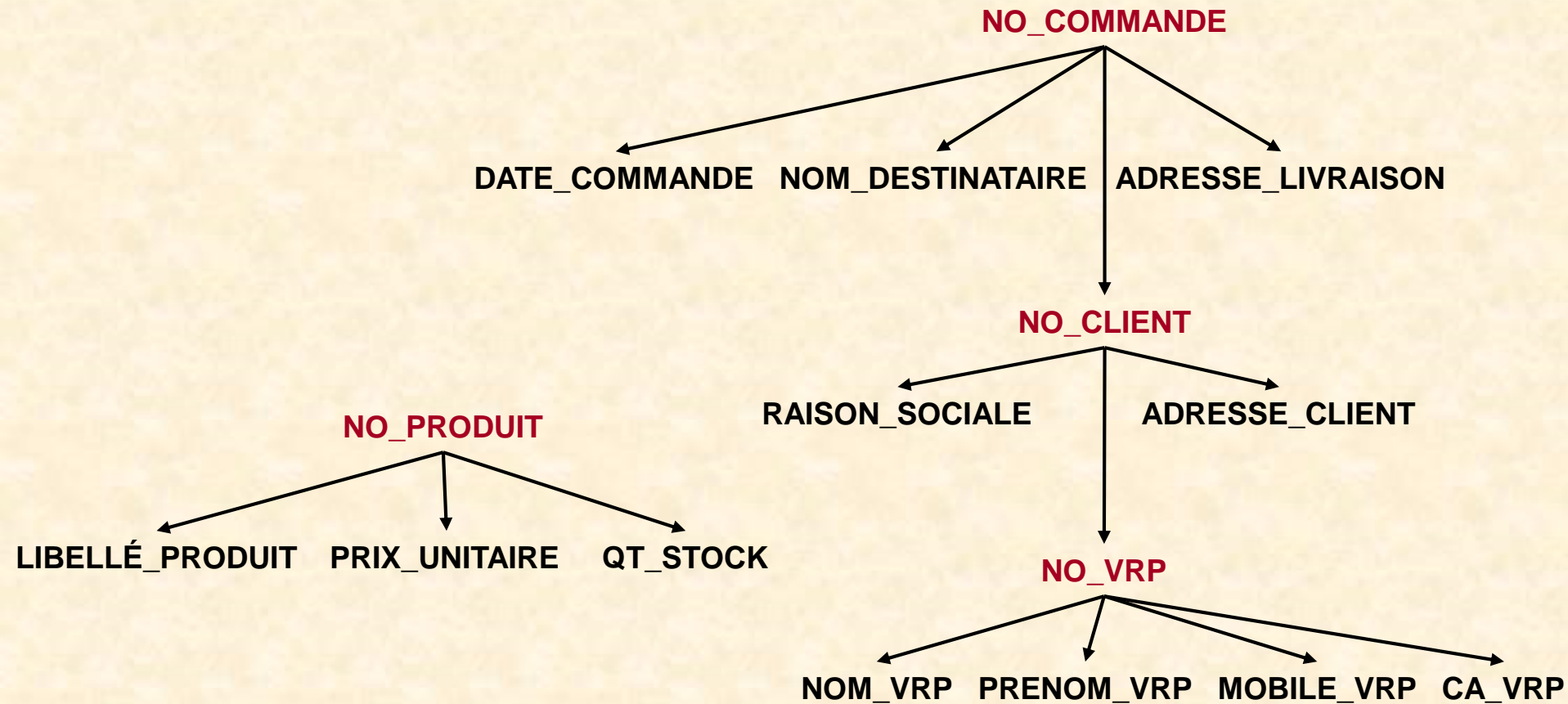
NO_VRP

Exemple de processus d'obtention d'un schéma rel.



- Exemple d'application
 - Étape 2 (suite) : rattachement des attributs déterminés aux attributs déterminants (de bas en haut)
 - cf. transparent suivant

Exemple de processus d'obtention d'un schéma rel.



Exemple de processus d'obtention d'un schéma rel.

□ Exemple d'application

□ Étape 3 : attributs non encore rattachés

□ JOUR_OUVRÉ

□ QT_COMMANDÉE

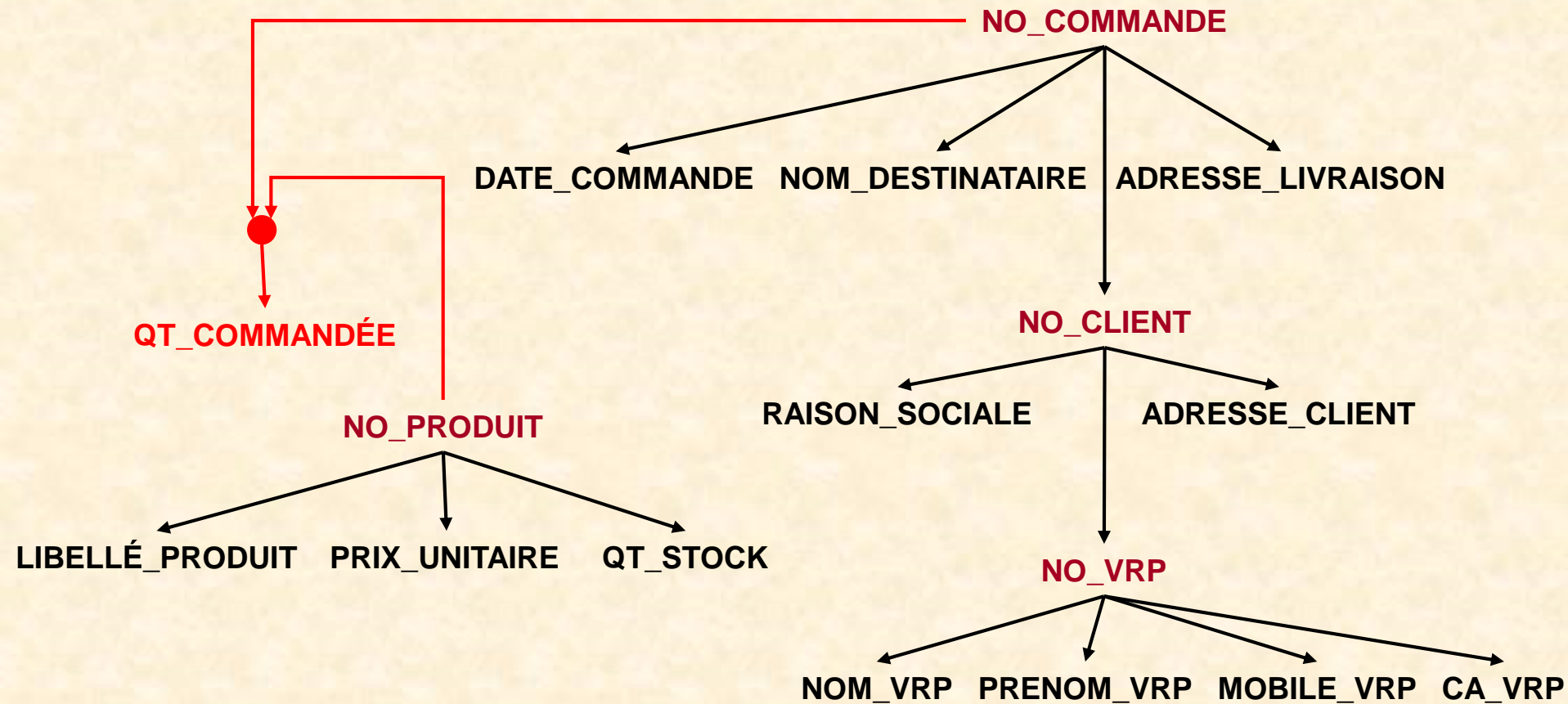
□ QT_PRODUIT_COMMANDÉ_JOUR

□ Étape 4

□ Pour les attributs identifiés à l'étape 3, recherche de DF composées en partie gauche d'attribut **S** ($N:1, N>1$) déterminants identifiés à l'étape 2

- (**NO_COMMANDE**, **NO_PRODUIT**) → QT_COMMANDÉE

Exemple de processus d'obtention d'un schéma rel.



Exemple de processus d'obtention d'un schéma rel.

□ Exemple d'application

□ Étape 5

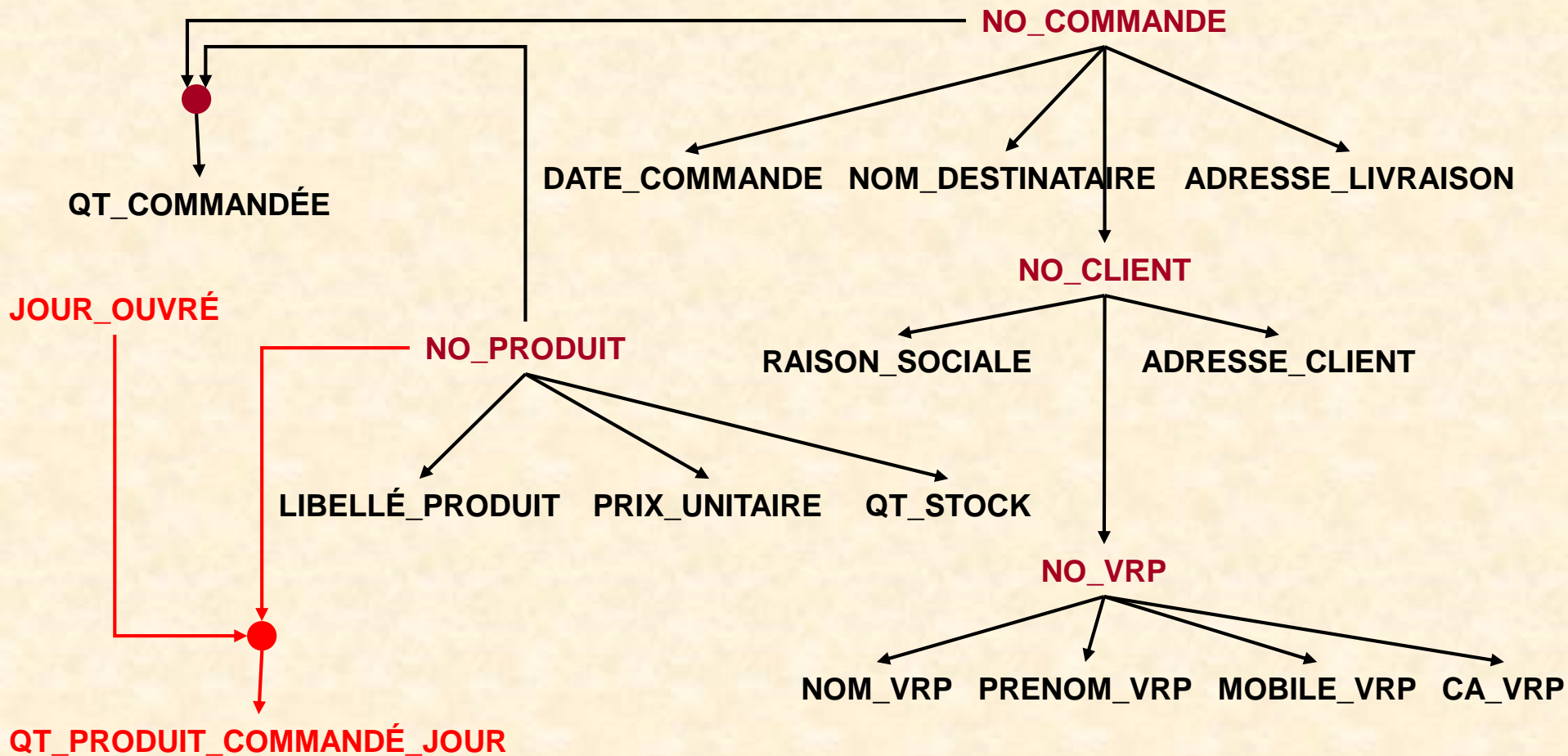
□ Pour les attributs non encore affectés

- JOUR_OUVRÉ
- QT_PRODUIT_COMMANDÉ_JOUR

□ Recherche des DF où l'un de ces attributs apparaît en partie gauche ($N:1, N>1$)

- (NO_PRODUIT, JOUR_OUVRÉ) → QT_PRODUIT_COMMANDÉ_JOUR

Exemple de processus d'obtention d'un schéma rel.



Exemple de processus d'obtention d'un schéma rel.

□ Schéma relationnel obtenu

COMMANDE (NO COMMANDE, NO CLIENT, DATE_COMMANDE, NOM_DESTINATAIRE, ADRESSE_LIVRAISON)

CLIENT (NO CLIENT, NO_VRP, RAISON_SOCIALE, ADRESSE_CLIENT)

VRP (NO_VRP, NOM_VRP, PRENOM_VRP, MOBILE_VRP, CA_VRP)

PRODUIT (NO PRODUIT, LIBELLÉ_PRODUIT, PRIX_UNITAIRE, QT_STOCK)

LIGNE_COMMANDE (NO COMMANDE, NO PRODUIT, QT_COMMANDÉE)

JOUR_OUVRÉ (JOUR_OUVRÉ)

QT_PRODUIT_COMMANDÉ_JOUR (NO PRODUIT, JOUR_OUVRÉ, QT_PRODUIT_COMMANDÉ_JOUR)

□ En quelle NF est ce schéma ?

Exemple de processus d'obtention d'un schéma rel.



- Schéma relationnel obtenu (suite)
 - Il y a deux attributs calculés
 - CA_VRP
 - QT_PRODUIT_COMMANDÉ_JOUR
 - De façon générale, l'on évite d'inclure dans le schéma des attributs qui peuvent être calculés à partir d'autres