

# BDR - 41803 – COURS 5

## Conception de BD réparties

Fragmentation

Réplication

2019

1

## Bases de Données Réparties

- Définition
- Conception
- Décomposition
- Fragmentation horizontale et verticale
- Outils d'interface SGBD
  - extracteurs, passerelles
- Réplication
- SGBD répartis hétérogènes

2

## BD réparties (1)

- Principe
  - Un site héberge une BD : accès local rapide, interne au site.
  - Accès global possible à des BD situées sur des sites externes
- Plusieurs niveaux d'intégration :
  - Accès distant (Remote Data Access)
  - Vues réparties
    - Extension du mécanisme de vues
    - Définir des vues sur plusieurs sites.
  - Médiateurs
  - BD réparties/fédérées



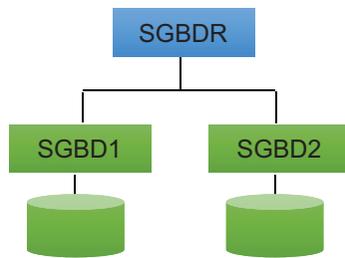
3

## BD Réparties (2)

- BD réparties :
  - Plusieurs BD sur plusieurs sites, mais une seule BD « logique ».
  - Fédérée : intègre des bases et des schémas existants
  - Répartie « pur » : conçue répartie. Pas d'accès locaux
- Les ordinateurs (appelés **sites**) communiquent via le réseau et sont faiblement couplés
  - pas de partage de MC, disque, au contraire de *BD parallèles*
- Chaque **site**
  - contient des données de la base,
  - peut exécuter des transactions/requêtes **locales** et
  - **participer** à l'exécution de transactions/requêtes **globales**

4

## SGBD réparti



Rend la répartition (ou distribution) *transparente*

- dictionnaire des données (catalogue, métabase) réparties
- traitement des requêtes réparties
- gestion de transactions réparties
- gestion de la cohérence et de la sécurité

5

## Paramètres à considérer

- Coût et **temps de communication** entre deux sites
  - Accès réseau (longue distance, WAN, MAN) beaucoup plus coûteux que accès disque
- Fiabilité : fréquence des pannes
  - des sites, du réseau (cf. P2P)
- Accessibilité aux données
  - accès aux données en cas de panne des sites, du réseau.
  - accès aux sites les moins encombrés, les plus puissants
- A lire : <https://read.acloud.guru/why-and-how-do-we-build-a-multi-region-active-active-architecture-6d81acb7d208>



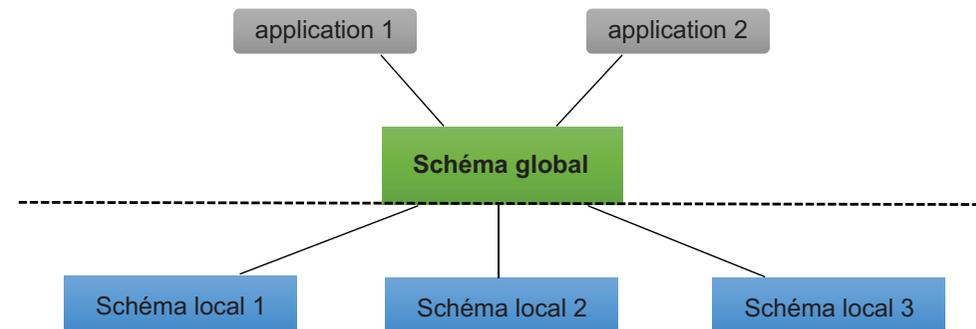
6

## Evaluation de l'approche BDR

- avantages
  - extensibilité
  - partage des données hétérogènes et réparties
  - performances avec le parallélisme
  - Disponibilité et localité avec la réplication
- inconvénients
  - administration complexe
  - complexité de mise en œuvre
  - distribution du contrôle
  - surcharge (l'échange de messages augmente le temps de calcul)

7

## Architecture de schémas



- indépendance applications / **bases locales**
- **schéma global** lourd à gérer

8

## Schéma global

### • Schéma conceptuel global

- description globale et unifiée de toutes les données de la BDR
  - Nom des relations avec leurs attributs
- Fournir l'indépendance à la répartition

### • Schéma de placement

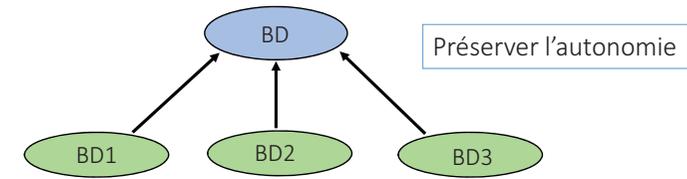
- règles de correspondance avec les données locales
  - Vues globales définies sur les relations locales (cf. global as view)
- Fournit l'indépendance à la localisation, la fragmentation et la duplication

- Le schéma global fait partie du dictionnaire de la BDR et peut être conçu comme une BDR (dupliqué ou fragmenté)

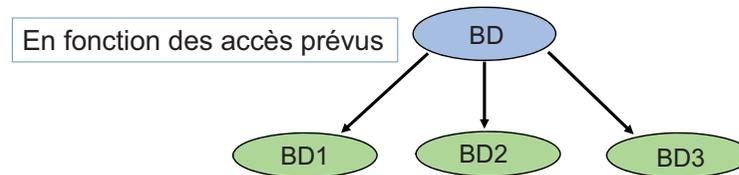
9

## Migration vers une BDR : 2 approches

**Intégration** logique des BD locales existantes (fédérée, médiateur)

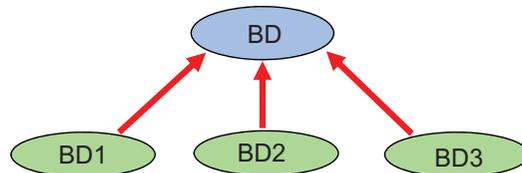


**Décomposition** en BD locales : répartie « pur »



10

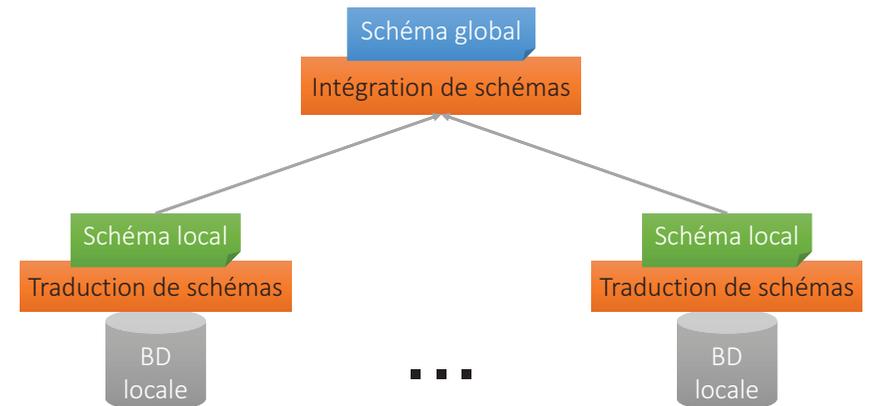
## Intégration de BD existantes



11

## Conception d'une BDR par intégration

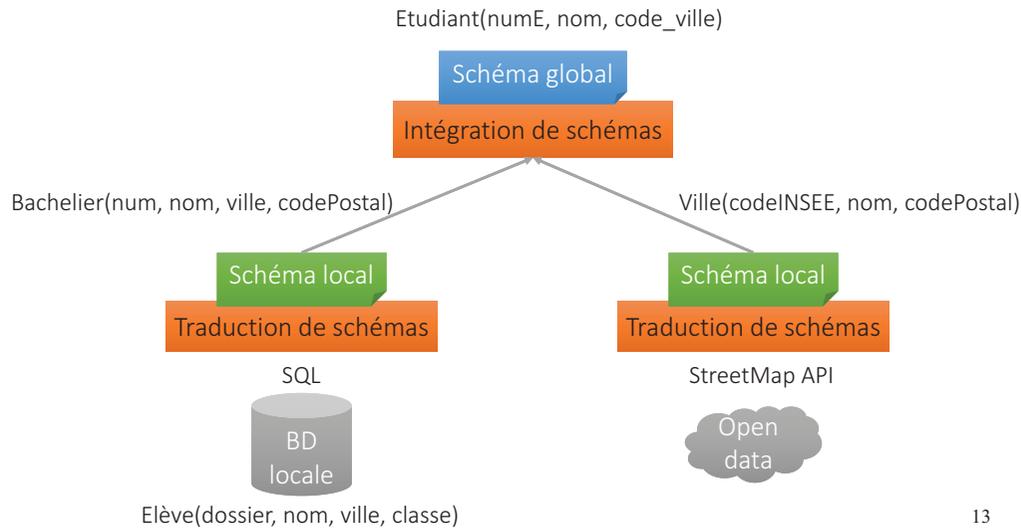
### Approche médiateur



12

# Conception d'une BDR par intégration

## Exemple



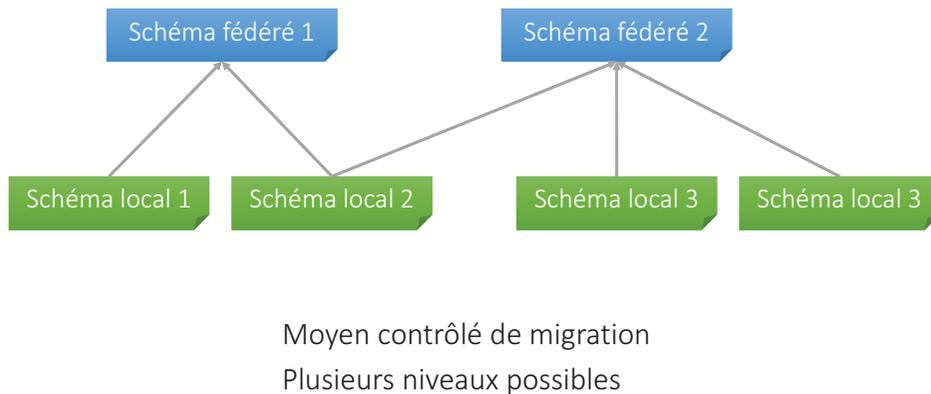
13

# Intégration de schémas

- 1. pré-intégration
  - Les schémas sont transformés pour les rendre plus homogènes
  - identification des éléments reliés (e.g. domaines équivalents) et établissement des règles de conversion (e.g. 1 inch = 2,54 cm)
  - Pbs : hétérogénéité des modèles de données, des puissances d'expression, des modélisations
- 2. comparaison
  - identification des conflits de noms (synonymes et homonymes) et des conflits structurels (types, clés, dépendances)
- 3. conformance
  - résolution des conflits de noms (renommage) et des conflits structurels (changements de clés, tables d'équivalence)
  - Définition de règles de traduction entre le schéma intégré et les schémas initiaux.

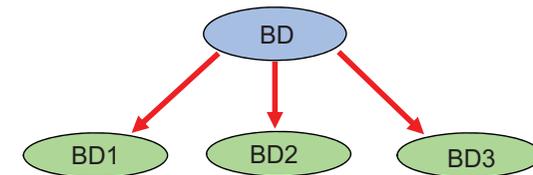
14

# Architecture fédérée



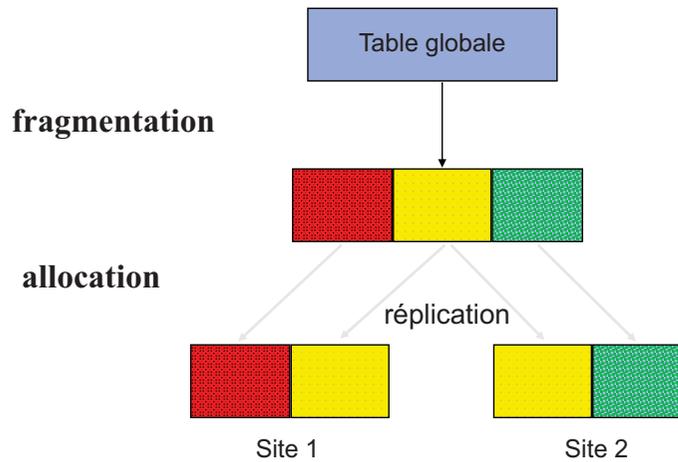
15

# Décomposition



16

## Conception par décomposition



17

## Objectifs de la décomposition

### Fragmentation

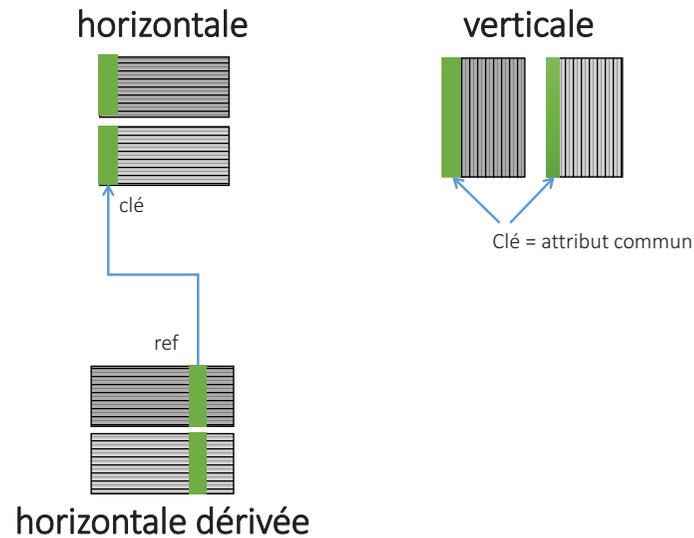
- Trois types : horizontale, horizontale dérivée, verticale
    - Possibilité de composer plusieurs fragmentations: mixte
  - Performances en favorisant les accès (et traitements) locaux
  - Equilibrer la charge de travail entre les sites (parallélisme)
  - Contrôle de concurrence plus simple pour les accès à un seul fragment
- Trop fragmenter : BD éclatée, nombreuses jointures réparties**

### Duplication (ou réplication)

- favoriser les accès locaux
  - augmenter la disponibilité des données
- Trop répliquer : surcoût de maintenir cohérence des répliques**

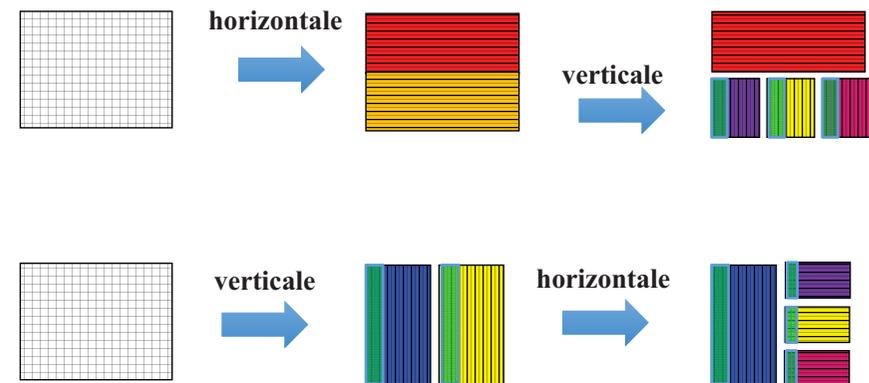
18

## Types de Fragmentation



19

## Fragmentation Mixte



20

## Fragmentation correcte

### Complète

- chaque élément de R doit se trouver dans un fragment

### Reconstructible

- on doit pouvoir recomposer R à partir de ses fragments (ressemble à décomposition de schéma vue en Li341 pour fragmentation verticale)

[Disjointe] /\*si on veut éviter réplication pour cohérence \*/

- chaque élément de R ne doit pas être dupliqué (sauf clé en cas de fragmentation verticale)

21

## Fragmentation Horizontale

### Fragments définis par sélection

$$\text{Client}_1 = \sigma_{\text{ville} = \text{'Paris'}} \text{Client}$$

$$\text{Client}_2 = \sigma_{\text{ville} \neq \text{'Paris'}} \text{Client}$$

Inférence : correcte

### Reconstruction par union

$$\text{Client} = \text{Client}_1 \cup \text{Client}_2$$

En SQL :

```
create view Client as
select * from Client1
union
select * from Client2
```

Client

nclient	nom	ville
C 1	Dupont	Paris
C 2	Martin	Lyon
C 3	Martin	Paris
C 4	Smith	Lille

Client1

nclient	nom	ville
C 1	Dupont	Paris
C 3	Martin	Paris

Client2

nclient	nom	ville
C 2	Martin	Lyon
C 4	Smith	Lille

22

## Fragmentation Horizontale : Exemples

### Fragments définis par sélection d'intervalles

Ville(numV, nom, cp, population)

$$V_1 = \sigma_{\text{population} < 10K} \text{Ville}$$

$$V_2 = \sigma_{10K \leq \text{population} < 50K} \text{Ville}$$

$$V_3 = \sigma_{50K \leq \text{population}} \text{Ville}$$

### Fragments définis par sélection sur plusieurs attributs

Personne(numP, nom, prenom, age, statut)

$$P_1 = \sigma_{\text{age} < 18 \text{ and } \text{statut} = \text{élève}} \text{Personne}$$

$$P_2 = \sigma_{\text{age} < 18K \text{ and } \text{statut} = \text{employé}} \text{Personne}$$

$$P_3 = \sigma_{\text{age} \geq 18} \text{Personne}$$

23

## Fragmentation Horizontale par sélection

- Fragmentation de R selon  $n$  prédicats
  - les prédicats  $\{p_1, \dots, p_n\}$  ex:  $\{a < 10, a > 5, b = 'x', b = 'y'\}$
- L'ensemble  $M$  des prédicats de fragmentation est :
  - $M = \{m \mid m = \bigwedge_{1 \leq k \leq n} p_k^* \}$  avec  $p_k^* \in \{p_k, \neg p_k\}$
  - Éliminer les  $m$  de sélectivité nulle ex:  $a > 10 \wedge a < 5$
  - Simplifier:  $a < 10 \wedge a \leq 5 \wedge b = 'x' \wedge b \neq 'y'$  devient  $a \leq 5 \wedge b = 'x'$
- Construire les fragments  $\{R_1, \dots, R_k\}$ 
  - Pour chaque  $m_i$ ,  $R_i = \sigma_{m_i}(R)$
- Minimalité
  - Ne pas avoir 2 fragments toujours lus ensemble
- Choisir les  $p_i$  des requêtes les plus fréquentes
- Ref biblio récente (2015) sur la fragmentation et le choix des fragments optimaux. Regroupement hiérarchique d'ensembles de nuplets issus de requêtes fréquentes.
  - Waterloo Univ : G. Aluç, M. T. Özsu, K. Daudjee and O. Hartig.
    - "Executing Queries over Schemaless RDF Databases", ICDE 2015 (Int'l Conf. on Data Engineering)

24

## Fragmentation Horizontale Dérivée

Fragments définis par **semi jointure**

```
Cde1 = select Cde.*
      from Cde, Client 1
      where Cde.nclient = Client1.nclient
```

$Cde_i = Cde \bowtie Client_i$  pour  $i$  dans  $\{1; 2\}$

Cde			
ncde	nclient	produit	qté
D 1	C 1	P 1	10
D 2	C 1	P 2	20
D 3	C 2	P 3	5
D 4	C 4	P 4	10

Cde1			
ncde	nclient	produit	qté
D 1	C 1	P 1	10
D 2	C 1	P 2	20

Cde2			
ncde	nclient	produit	qté
D 3	C 2	P 3	5
D 4	C 4	P 4	10

**Reconstruction par union**

$Cde = Cde_1 \cup Cde_2 = \bigcup_i Cde_i$

25

## Fragmentation Horizontale Dérivée Méthode générale

- Commencer par recenser les fragments horizontaux déjà définis
- Repérer des clés étrangères dans la relation à fragmenter
- Semi-jointure(s)

Exemple avec 2 semi-jointures

- $Personne(numP, nom, premon)$ 
  - 4 fragments pour  $a=[0,18,25,60,100]$
  - $P_i = \sigma_{ai-1 \leq age < ai} Personne$
- $Sport(numS, type)$ 
  - 2 fragments pour  $t=[indiv, collectif]$
  - $S_i = \sigma_{type = ti} Sport$
- $Inscrit(numP*, sport*, date)$ 
  - $4*2 = 8$  fragments:  $Inscrit_i = Inscrit \bowtie P_i \bowtie S_i$

26

## Propriétés de la fragmentation horizontale dérivée

R: fragmentation horizontale  $\rightarrow$  fragments  $R_i$

S: fragmentation horizontale dérivée  $\rightarrow$  fragments  $S_i = S \bowtie_A R_i$

- **Complète**
  - Chaque tuple de S doit joindre avec au moins un tuple de R
  - $\forall s \in S, \exists t \in S_i, s = t$
- **Disjointe**
  - $\forall i, j \text{ tq } i \neq j, S_i \cap S_j = (S \bowtie R_i) \cap (S \bowtie R_j) = S \bowtie (R_i \cap R_j) = \emptyset$
  - Rappel:  $R_1 \cap R_2 \Leftrightarrow R_1 \bowtie R_2$
- **Reconstructible**
  - $\bigcup_i S_i = (S \bowtie R_1) \cup (S \bowtie R_2) \cup \dots \cup (S \bowtie R_n) = S \bowtie (\bigcup_i R_i) = S$
- $\Rightarrow$  contrainte d'intégrité référentielle
  - A = clé de R
  - S.A référence R.A
    - $\forall s \in S, \exists r \in R, s.A = r.A$

## Fragmentation Verticale

Fragments définis par **projection**

$Cde1 = \pi_{ncde, nclient} Cde$

$Cde2 = \pi_{ncde, produit, qté} Cde$

Cde			
ncde	nclient	produit	qté
D 1	C 1	P 1	10
D 2	C 1	P 2	20
D 3	C 2	P 3	5
D 4	C 4	P 4	10

**Reconstruction par jointure**

$Cde = Cde1 \bowtie Cde2$

En SQL :

```
create view Cde as
select * from Cde1, Cde2
where Cde1.ncde = Cde2.ncde
```

Cde1		Cde2		
ncde	nclient	ncde	produit	qté
D 1	C 1	D 1	P 1	10
D 2	C 1	D 2	P 2	20
D 3	C 2	D 3	P 3	5
D 4	C 4	D 4	P 4	10

28

# Fragmentation Verticale

Comment définir une fragmentation verticale ?

- Affinité des attributs : mesure la proximité sémantique des attributs (combien « ils vont ensemble »)
  - Soit par connaissance de l'application,
  - soit par analyse des requêtes (on mesure combien de fois deux attributs donnés ont été interrogé ensemble)
  - Résultat sous forme de matrice d'affinité
- 2 approches : regroupement, partitionnement (grouping – splitting)
  - Idem que pour optimisation de schéma relationnel SPI, SPD
- Algorithme de regroupement des attributs bien adapté
  - BEA : bond energy algorithm (Mc Cormick et al. 72) :  $O(n^2)$
  - insensible à l'ordre de départ des attributs
  - Part des attributs individuels et effectue des regroupements de groupes
- Algorithme de partitionnement :
  - Part d'une relation et observe le bénéfice qu'on peut tirer à partitionner

29

# Matrice d'affinité des attributs

Matrice A

$a_{ij}$  = affinité de  $A_i$  avec  $A_j$

ex: nb de requêtes qui accèdent  $A_i$  et  $A_j$

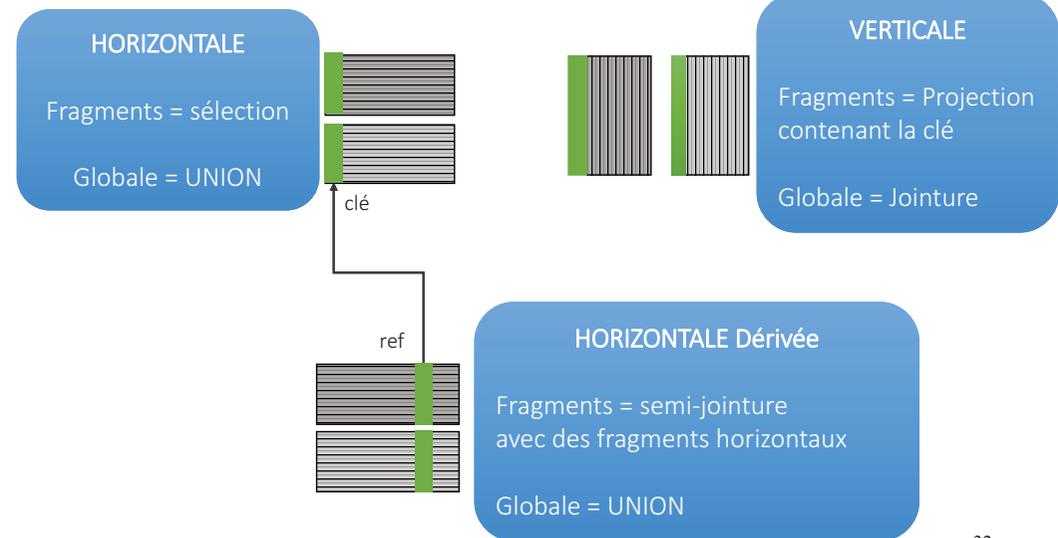
	A1	A2	A3	A4
A1	45	0	45	0
A2		80	5	75
A3			53	3
A4				78

# Matrice d'affinité regroupement des attributs

Matrice A

	A1	A3	A2	A4
A1	45	45	0	0
A3		53	5	3
A2			80	75
A4				78

# Résumé des Fragmentations



## Allocation des Fragments aux Sites

### Non-répliquée

- partitionnée : chaque fragment réside sur un seul site

### Dupliquée

- chaque fragment sur un ou plusieurs sites
- maintien de la cohérence des copies multiples : coûteux
- (le fameux) Compromis Lecture/écriture:**  
+ le ratio Lectures/màj est  $> 1$ , + la duplication est avantageuse

33

## Allocation de Fragments

Problème: Soit

$F$  un ensemble de fragments

$S$  un ensemble de sites

$Q$  un ensemble d'applications et leurs caractéristiques

trouver la distribution "optimale" de  $F$  sur  $S$

Optimum

- coût minimal de communication, stockage et traitement
- Performance = temps de réponse ou débit

Solution

- allouer une copie de fragment là où le bénéfice est supérieur au coût

34

## Exemple d'Allocation de Fragments

Client1

nclient	nom	ville
C 1	Dupont	Paris
C 3	Martin	Paris

Client2

nclient	nom	ville
C 2	Martin	Lyon
C 4	Smith	Lille

Cde1 = Cde  $\triangleright$  Client1

ncde	client	produit	qté
D 1	C 1	P 1	10
D 2	C 1	P 2	20

Site 1

Cde2

ncde	client	produit	qté
D 3	C 2	P 3	5
D 4	C 4	P 4	10

Site 2

35

## Exemple

Trois universités parisiennes (Jussieu, Sorbonne, Dauphine) ont décidé de mutualiser leurs équipements sportifs (locaux) et les entraîneurs. La gestion commune est effectuée par une **base de données répartie**, dont le schéma **global** est le suivant :

**PROF** (Idprof, nom, adresse, tél, affectation, salaire)

**ETUDIANT** (Idetu, nom, adresse, assurance, police, université, équipe)

**LOCAUX** (Idlocal, adresse, université)

**EQUIPE** (équipe, sport, niveau)

**HORAIRE** (Idlocal, équipe, jour, heure\_début, heure\_fin, prof)

- Chaque université rémunère ses profs en envoyant un chèque à leur adresse, mais aussi elle doit pouvoir contacter tout prof qui utilise ses locaux.
- Chaque équipe correspond à un sport. La plupart des équipes ont droit à un (seul) créneau (jour, heure) dans un des locaux communs pour leur entraînement. Cependant, pour le sport «cyclisme», il n'y a pas besoin de locaux.
- Chaque université gère évidemment ses propres étudiants, ainsi que ses locaux et les créneaux correspondants.
- Les équipes ne sont associées à aucune université en particulier. Cependant, pour des questions d'assurance, chaque université doit aussi gérer les étudiants qui utilisent ses locaux. Pour le cyclisme, c'est Dauphine qui en a la charge.
- Les relations globales sont **fragmentées et réparties sur les différents sites**.

36

## Données réparties avec Oracle : Database link

Lien à une table dans une BD distante spécifié par :

- nom de lien
- nom de l'**utilisateur** et **password**
- Infos de **connexion** (protocole client-serveur d'oracle)

Exemple de syntaxe :

```
create database link Site2  
connect to E1234 identified by "E1234" using 'ora10';
```

Create synonym Emp2 for Emp@Site2;

*Ou*

Create view Emp2 as select \* from Emp@Site2;



Vue  
répartie