



Gestion de Données Réparties



Claudia Roncancio

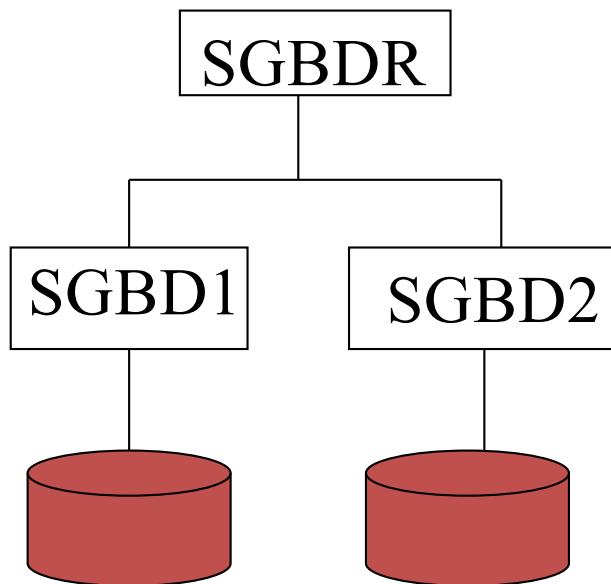
Claudia.Roncancio@grenoble-inp.fr

+ Définitions BD et SGBD répartis

- Une *base de données répartie* est une collection de BD logiquement reliées, chacune gérée par un SGBD local à un site, en générale distribuées sur un réseau
- Un **SGBD réparti** fourni des mécanisme d'accès rendant la répartition « transparente » aux utilisateurs :
 - Dictionnaire des données réparties
 - Traitement des requêtes réparties
 - Gestion de transactions reparties
 - Communication de données inter-sites
 - Gestion de cohérence et de sécurité

+ SGBD réparti

3



Logiquement réparti
SGBDR, SGBD1, SGBD2

Physiquement réparti
Site 1 : SGBDR, SGBD1,
Site 2 : SGBDR, SGBD2

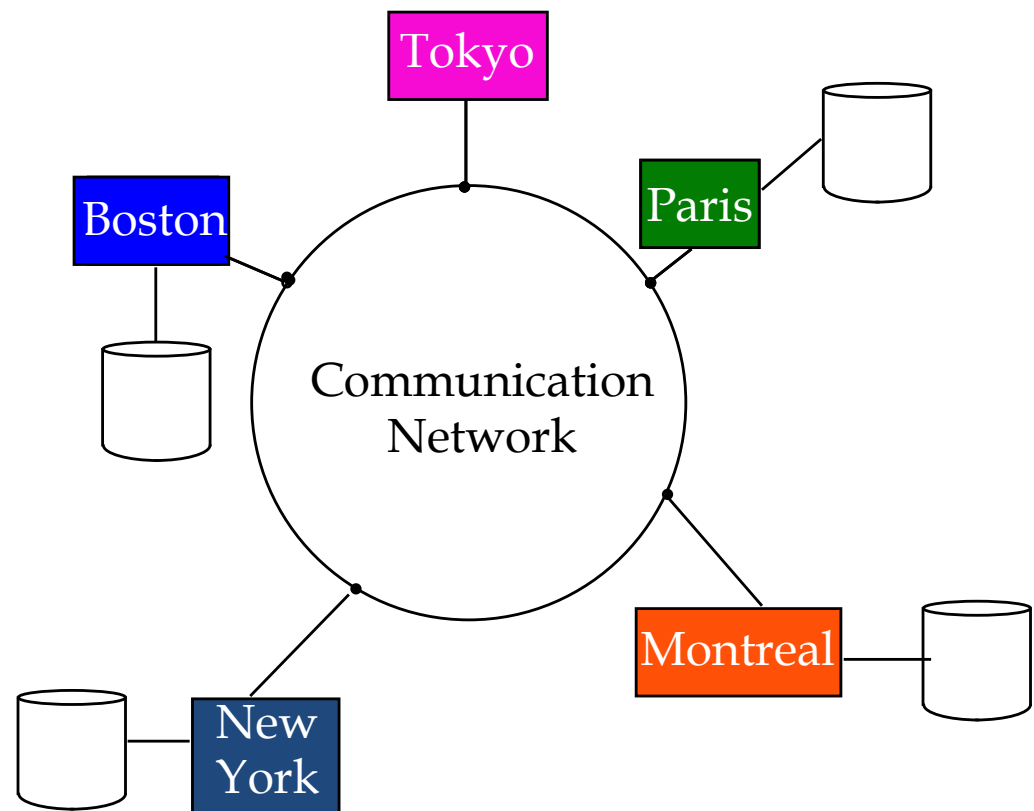
+ Example

EMP(ENO, ENAME, TITLE)

PROJ(PNO, PNAME, BUDGET, LOC)

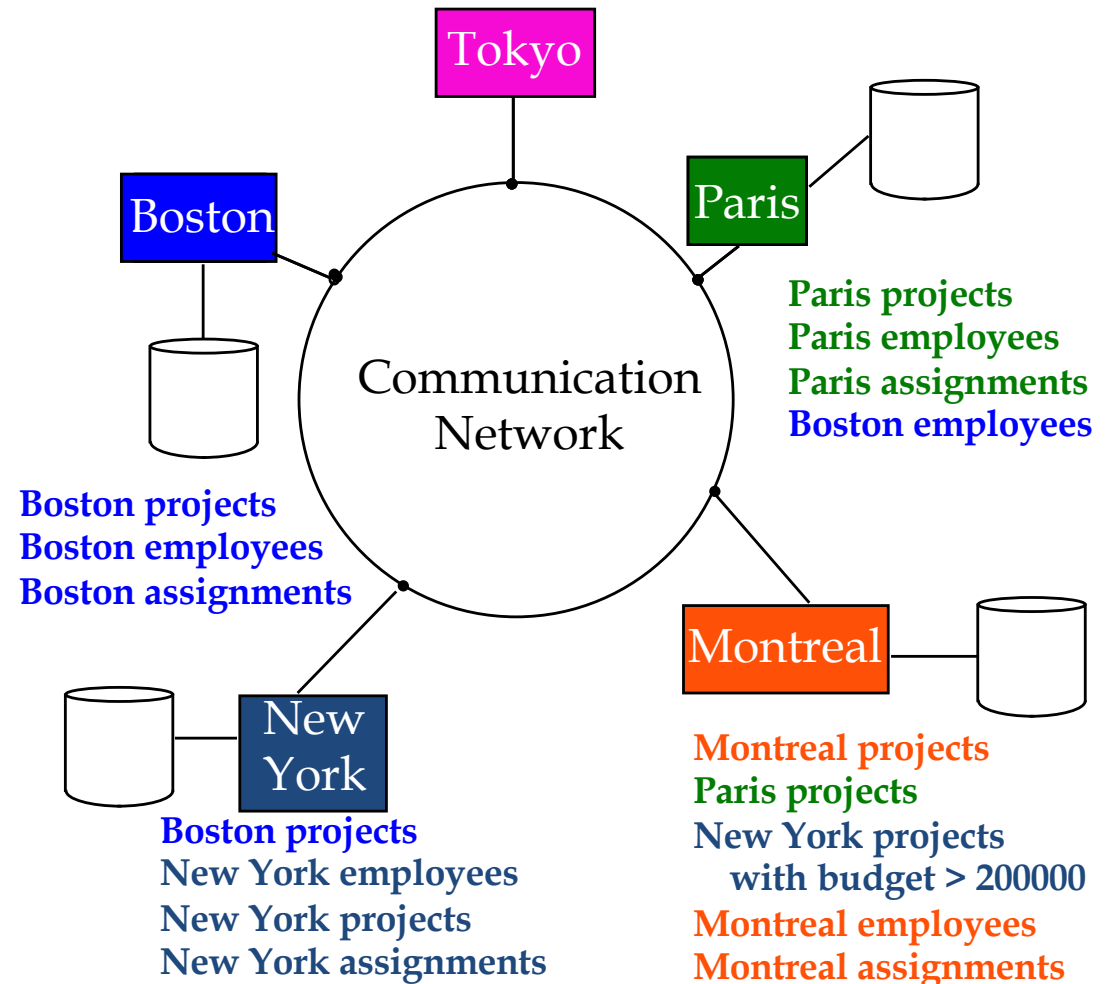
PAY(TITLE, SAL)

ASG(ENO, PNO, DUR, RESP)



+ Transparent Access

```
SELECT  ENAME, SAL  
  
FROM    EMP, ASG, PAY  
  
WHERE   DUR > 12  
  
AND     EMP.ENO = ASG.ENO  
  
AND     PAY.TITLE = EMP.TITLE
```



+ Niveaux de transparence

- Indépendance des données

logique / physique

- Réseaux

- Fragmentation, Duplication

+ Alternatives dans l'architecture d'un SGBDR

■ Autonomie

- Contrôle / Conception, communication, exécution

■ Distribution

- Clients/Serveurs, clusters, cloud, Pair-à-pair (P2P)

■ Hétérogénéité

- Matériel, réseaux
- Modèle de données, langages, système
- Sémantique



Fragmentation,
allocation,
sharding



+ Approches de conception d'une BD Distribuée

- Approche descendante (décomposition)
 - Conception du schéma conceptuel global
 - Distribution pour obtenir des schémas conceptuels locaux
 - Fragmentation
 - Affectation aux sites - Allocation
- Approche ascendante
 - Intégration de bases de données existantes
 - Hétérogénéité

+ Exemple

12

- Relation Employé (nss, nom, loc, ...)
- Relation Taux (pays, valeur, ...)
- 12 bureaux d'environ la même taille
 - 6 à Paris, 4 à Marseille et 2 à Lyon
- Quelles requêtes ? Transactions ?



Conception d'une BDD

14

- De la conception en centralisé
 - schéma conceptuel global
 - schéma physique
- En distribué
 - définition des fragments (pour quoi?)
 - conception « physique » : placement des fragments, stockage, chemins d'accès.

+ Comment fragmenter ?

- Grain / degré de fragmentation
 - Trop peu de fragments - faible concurrence
 - Trop de fragments - surcoût dans la reconstruction des relations
- Possibilités de fragmentation d'une relation
 - **Horizontale** - basée sur des sélections
 - **Verticale** - basée sur des projections
 - **Hybride**

+ Et le placement des fragments ?

- Chaque fragment sur un seul site
 - copie unique, BD partitionnée
- Duplication de fragments
 - (+) performances des requêtes et disponibilité
 - (-) coût des mises à jour et contrôle de concurrence plus complexe
- Fréquemment : duplication partielle
- Applications pour la duplication totale

+ Objectifs généraux

18

■ Fragmentation

- Favoriser les accès locaux
- Distribuer la charge de travail

■ Duplication

- Favoriser les accès locaux
- Augmenter la disponibilité des données

Informations utiles

- Données : taille des données, CI
- Applications : FAQ et où
- Capacités des sites / réseaux

+ Fragmentation Horizontale

- Fragments définis par sélection

Ex : Clients(NClient, Nom, Ville)

Client1 = $\sigma_{\text{Ville} = \text{Paris}}(\text{Client})$

Client2 = $\sigma_{\text{Ville} \neq \text{Paris}}(\text{Client})$

- Reconstruction par union des fragments

Ex : Client = Client1 \cup Client2

Fragmentation Horizontale – Exemple

Relation Client

| NoClient | Nom | Ville |
|----------|--------|----------|
| C1 | Dupont | Paris |
| C2 | Martin | Grenoble |
| C3 | Martin | Paris |
| C4 | Talon | Lille |

Fragmentation Horizontale - Suite exemple

| NoClient | Nom | Ville |
|----------|--------|-------|
| C1 | Dupont | Paris |
| C3 | Martin | Paris |

Client1

| NoClient | Nom | Ville |
|----------|--------|----------|
| C4 | Talon | Lille |
| C2 | Martin | Grenoble |

Client2

+ Fragmentation Horizontale Dérivée

- Fragments définis par (semi) jointure
- Ex : Commande(NC, NClient, Produit,Qté)

Commande1 = Commande α Client1

Commande2 = Commande α Client2

- Reconstruction par union des fragments

Ex : Commande = Commande1 \cup Commande2

Fragmentation Horizontale Dérivée - Exemple

Relation Commande

| NC | NClient | Produit | Qte |
|-----|---------|---------|-----|
| Co1 | C1 | P1 | 10 |
| Co2 | C1 | P2 | 200 |
| Co3 | C2 | P3 | 30 |
| Co4 | C4 | P4 | 5 |

Fragmentation Horizontale Dérivée – Suite exemple

Relation Commande1

| NC | NClient | Produit | Qte |
|-----|---------|---------|-----|
| Co1 | C1 | P1 | 10 |
| Co2 | C1 | P2 | 200 |

Relation Commande2

| NC | NClient | Produit | Qte |
|-----|---------|---------|-----|
| Co3 | C2 | P3 | 30 |
| Co4 | C4 | P4 | 5 |

+ Fragmentation Verticale

- Fragments définis par projection

Ex : Commande(NC, NClient, Produit, Qté)

CommandeA = $\pi_{NC, NClient}$ (Commande)

CommandeB = $\pi_{NC, Produit, Qté}$ (Commande)

- Reconstruction par jointure

Ex : Commande = CommandeA \bowtie CommandeB

- Utile si forte affinité des attributs

Fragmentation Verticale - Exemple

Relation Commande

| NC | NClient | Produit | Qte |
|-----|---------|---------|-----|
| Co1 | C1 | P1 | 10 |
| Co2 | C1 | P2 | 200 |
| Co3 | C2 | P3 | 30 |
| Co4 | C4 | P4 | 5 |

Fragmentation Verticale – Suite

Exemple

| NC | NCClient |
|-----|----------|
| Co1 | C1 |
| Co2 | C1 |
| Co3 | C2 |
| Co4 | C4 |

Relation CommandeA

Relation CommandeB

| NC | Produit | Qte |
|-----|---------|-----|
| Co1 | P1 | 10 |
| Co2 | P2 | 200 |
| Co3 | P3 | 30 |
| Co4 | P4 | 5 |

+ Propriétés de la fragmentation

■ Reconstruction

- Possible avec les opérateurs de l'algèbre relationnelle
- Critères de fragmentation horizontale simples ou complexes

■ Complétude

- Horizontale : chaque n-uplet de R est dans un fragment
- Verticale : chaque attribut est dans un fragment
- Pas de perte d'information
- Similaire à la normalisation

Disjoint vs Non disjoint ?

+ Comment créer des fragments disjoints et assurer la reconstruction ?

- Génération automatique ou « manuelle »

Cas selon le type de fragmentation

- Prédicats de FAQ, forme normale conjonctive, accès « homogènes » des données d'un fragment,

+ Analyse Fragmentation Horizontale Dérivée

- R est fragmentée en R_1, \dots, R_k
- S est fragmentée en $S \bowtie R_1, \dots, S \bowtie R_k$

Conditions pour que ça marche :

Reconstruction : $S = \bigcup (S \bowtie R_i)$

Disjonction : $(i \neq j) \implies (S \bowtie R_i) \cap (S \bowtie R_j) = \emptyset$

Conception : association 1-n entre entités R et S

Intégrité : S a une clé étrangère de R

... Condition suffisante pour reconstruction et disjonction

+ Analyse Fragmentation Verticale

Emp(Enum, Nom, Ville, Sal)
Emp1(Enum, Nom, Ville) Emp2(Enum, Sal)

Complétude et reconstruction

- Jointure sans perte d'information
- Condition suffisante : répéter la clé dans chaque fragment
- Autre choix : utiliser une dépendance fonctionnelle pour la décomposition (cf normalisation)

Disjonction : seulement répéter la clé

+ Comment fragmenter verticalement ?

- Affinité des attributs

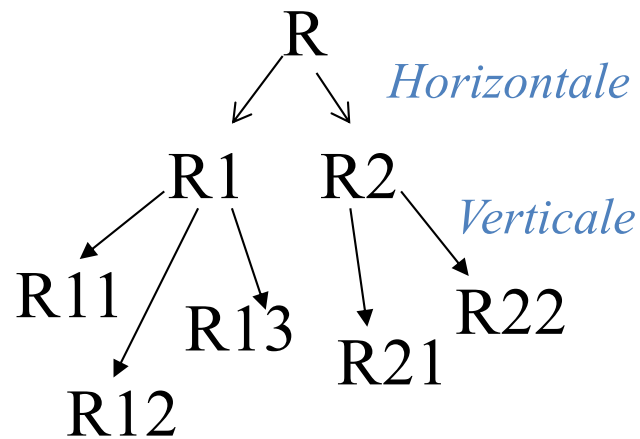
- Heuristiques :

- *Clustering* - rapprochement en partant de d'un attribut par fragment

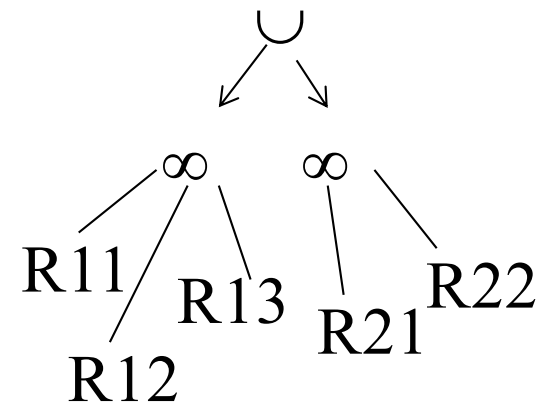
- *Splitting* - décomposition en partant d'un fragment unique pour obtenir une partition selon l'affinité des attributs

+ Fragmentation Hybride

- Fragmentation horizontale suivit de verticale ou vice-versa



Reconstruction



+ Allocation

35

- Problème d'optimisation très difficile
- Minimiser coûts: stockage, traitement, communication
- Maximiser performances: temps de réponse, débit du système
- Modèle coût / performances
- Approche simple considérer les com./accès

+ Exemple Client + Commandes

■ $\text{Client1} = \sigma_{\text{Ville} = \text{Paris}} (\text{Client})$

$\text{Client2} = \sigma_{\text{Ville} \neq \text{Paris}} (\text{Client})$

$\text{Commande1} = \text{Commande} \alpha \text{Client1}$

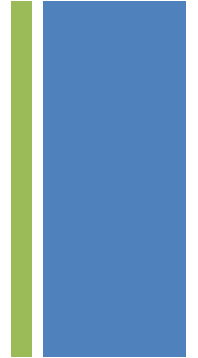
$\text{Commande2} = \text{Commande} \alpha \text{Client2}$

■ Allocation

@Site1 : Client1, Commande1

@Site2 : Client2, Commande2

+ Fragmentation / sharding

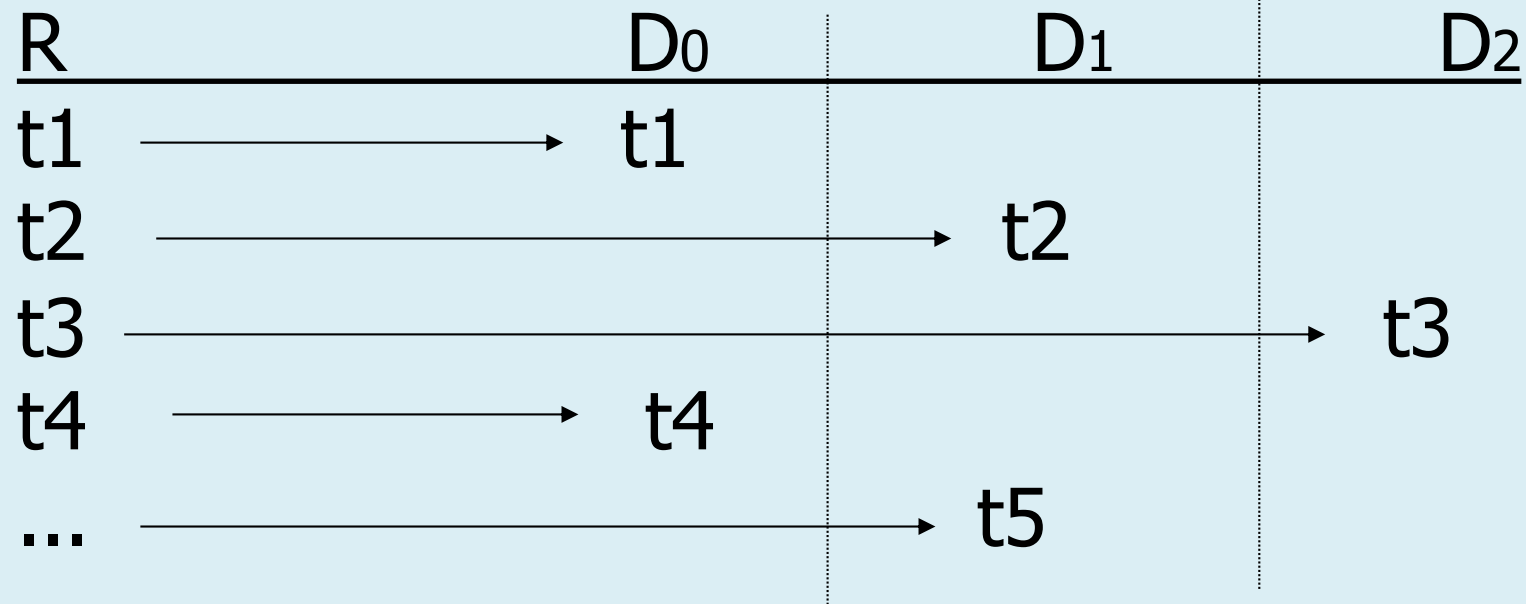


- A partition forms a « shard »
- Fragmentation based on FAQ or known access patterns
- Automatic partitioning, sharding?
- Sharding is often related to shared-nothing architectures

Three common horizontal partitioning techniques

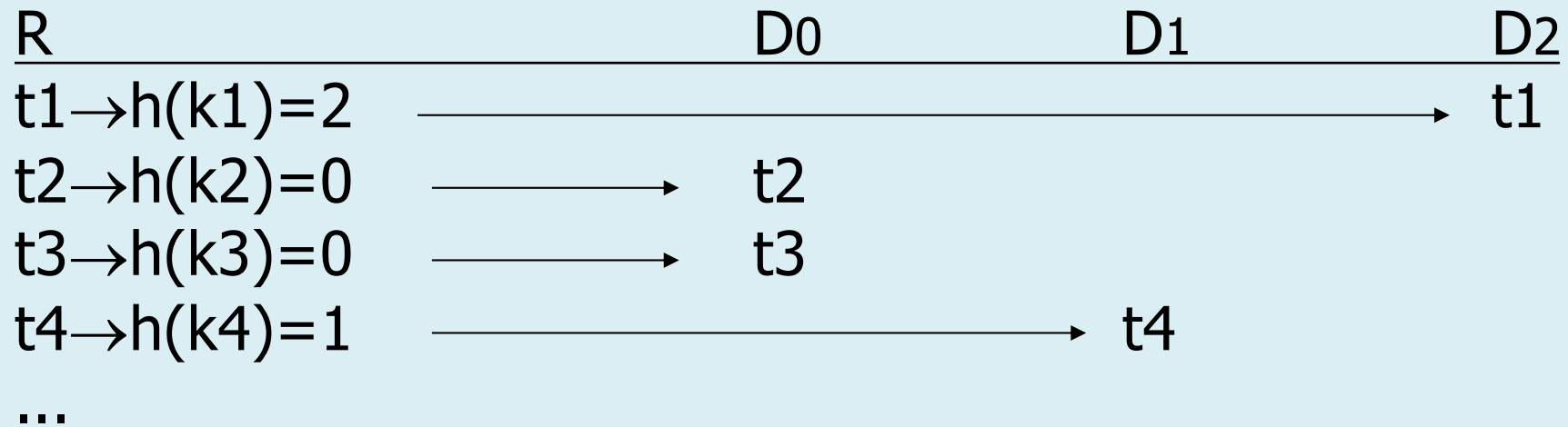
- Round robin
- Hash partitioning
- Range partitioning

Round robin



- Evenly distributes data
- Good for scanning full relation
- Not good for point or range queries

Hash partitioning



Good for point queries on key; also for joins

Not good for range queries; point queries not on key

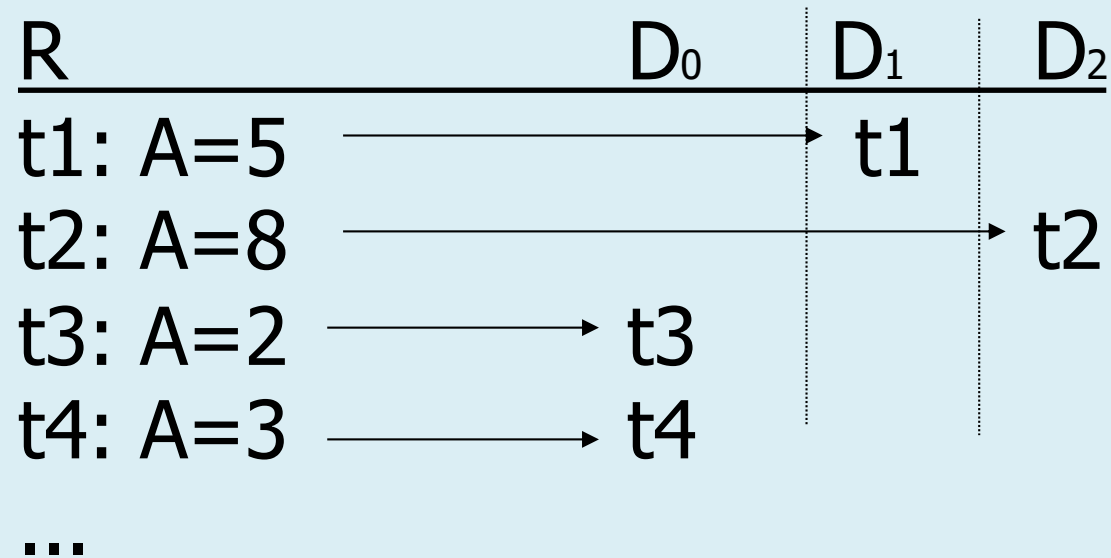
If a good hash function, even distribution

Range partitioning

partitioning
vector

| | |
|---|---|
| 4 | 7 |
|---|---|

V₀ V₁



- Good for some range queries on A
- Need to select good vector: else unbalance
 - data skew
 - execution skew

+ Conclusion

■ Fragmentation

- Décomposition / reconstruction
- Prises en compte des requêtes & CI
- Horizontale simple / dérivée, verticale, hybride
- Propriétés : reconstruction, complétude, disjonction

■ *Sharding*

- Approches automatiques

■ Allocation / duplication

- Modèle de coût
- Approche itérative

+ Bibliographie

- T. Ozsú, P. Valduriez, Principles of Distributed DB Systems, Prentice Hall
- Notes de S. Abiteboul
- H. Garcia Molina, J. Ullman, J. Widom, Database systems, the complete book, Pearson International Edition